NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

COMPUTER-AIDED COURSE ENROLLMENT SYSTEM FOR COMPUTER SCIENCE CURRICULUM OFFICE

by

Erkan Ertugrul

March, 1995

Thesis Advisor: Man-Tak Shing

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1995	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE COMPUTER-AIDED COURSE ENROLLMENT SYSTEM FOR COMPUTER SCIENCE CURRICULUM OFFICE 6. AUTHOR(S) Ertugrul, Erkan	5. FUNDING NUMBERS
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT The major problem addressed in this study is to automate the "Course Enrollment Process" in Computer Science Curriculum Office, thus to reduce the time spent in this process and increase the reliability and efficiency of the current system. The approach taken was to do a requirement analysis, design an automated system by using Yourdon's Structured Analysis Method and implement the design in C++ programming language. The resulting program that we wrote enables the curriculum staff to keep information about students, courses and tracks, and generate the enrollment list and send messages to the students. It also enables the students to prepare their matrices, sign up for courses, get course and track information and send messages to the curricular officer through the application. The curriculum staff has reviewed the program and thought that it could be used in the real-world environment. But, it was not put into service in the curriculum. A future study can install the program to the computers in the office and test how effective the program is.

14. SUBJECT TERMS Course Enrollment Process, Computer Science, Software Development, Software Design, Software Engineering, Windows Programming	15. NUMBER OF PAGES 219
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

COMPUTER-AIDED COURSE ENROLLMENT SYSTEM FOR COMPUTER SCIENCE CURRICULUM OFFICE

by

Erkan Ertugrul
First Lieutenant, Turkish Army
B.A., Turkish Military Academy, 1990

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 1995

Author: Ertugrul
Erkan Ertugrul

Approved by:
Man-Tak Shing, Thesis Advisor

Frank Kelbe, Second Reader

Ted Lewis
Ted Lewis, Chairman,
Department of Computer Science

ABSTRACT

The major problem addressed in this study is to automate the "Course Enrollment Process" in Computer Science Curriculum Office, thus to reduce the time spent in this process and increase the reliability and efficiency of the current system. The approach taken was to do a requirement analysis, design an automated system by using Yourdon's Structured Analysis Method and implement the design in C++ programming language. The resulting program that we wrote enables the curriculum staff to keep information about students, courses and tracks, and generate the enrollment list and send messages to the students. It also enables the students to prepare their matrices, sign up for courses, get course and track information and send messages to the curricular officer through the application. The curriculum staff has reviewed the program and thought that it could be used in the real-world environment. But, it was not put into service in the curriculum. A future study can install the program to the computers in the office and test how effective the program is.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

A. HISTORY

The idea of automating the "Course Enrollment Process" in the Computer Science Curriculum Office was first introduced to the students of Naval Postgraduate School students as a term project for the "CS3460 Software Methodology" class, taught by Prof. Man-Tak Shing in Fall-1993 quarter. The author of this thesis was one of the students attending this particular class.

Because of the time limitations, the students were asked to design the whole system, but to concentrate on a specific portion as far as the programming part of the project was concerned. Thus, the project was not accomplished to the full extent by any of the students in the class.

A quarter later, when Prof. Shing was making his presentation about his research areas to the students in the Computer Science Curriculum, he announced his willingness to supervise a student to complete the desing and development of the "Automation of the Course Enrollment Process" as a Master's Thesis project. Then, the author of this thesis, started working on this project.

B. RESEARCH QUESTIONS

1. Primary Research Question

How can the "Course Enrollment Process" of the Computer Science Curriculum Office be automated?

2. Secondary Research Questions

- What requirements and constraints does the "Course Enrollment Process" have?
- Will these requirements be satisfiable?
- Which tools and what language can be used for the implementation of the application?

C. OBJECTIVES

- To enable the Computer Science Curriculum Office staff to perform their tasks concerning the "Course Enrollment" in a more efficient and easier way.
- To reduce the amount of paper work in the curriculum office.
- To provide the curriculum staff with better, more intuitive and user-friendly tools to perform their jobs in keeping records of the students, the courses offered and the students' course matrices.
- To provide a better and more effective means of communication between the curriculum staff and the students.
- To enable the Computer Science Curriculum students to preparing their matrices and obtaining information about the tracks and courses in a more efficient way.
- To reduce the course enrollment related errors that result in loss of time and resources.

D. ORGANIZATION OF THE STUDY

Chapter II discusses how the process to be automated was designed. It also gives information about the reasons behind the design decisions made.

Chapter III presents what tools and language were selected to be used to write the code for the application and the reasons behind selecting these tools. The problems encountered and the solutions found while writing the program are also discussed.

Chapter IV concludes the study and gives recommendations for the future studies.

II. DESIGN OF THE SYSTEM

A. THE METHOD USED TO COLLECT INFORMATION ABOUT SYSTEM DESIGN

A close contact has been maintained between the author of this thesis, the curriculum staff, and the thesis advisor. Initially, a number of interviews were made with the curriculum staff concerning the current system being used and their expectations from an automated system. During both the design and the programming phase of the study, the curriculum staff were given presentations about what has been achieved so far and were consulted for their feedback about the growing project.

Since the author of this thesis was also a student in the Computer Science Curriculum, he was not only the system analyst and developer, but he was also a user of the system. For that reason, the task of the system analyst to understand the users' opinions and desires for the planned automation has been achieved easier than it is generally expected to be. A very effective coordination between the designer and the users of the system was sustained.

In the following sections, we present the result of our analysis using Yourdon's Structured Analysis Method.

B. REQUIREMENT ANALYSIS

1. Goals

The overall application software is going to be used by the office staff and the students of the Computer Science Curriculum . The students from other curriculums who are taking classes from the Computer Science Department are not intended users of the system. The application program is intended to increase the efficiency of the course enrollment system by giving the students the capability to enroll for courses in an automated way and by reducing the amount of the paper work that curriculum personnel must generate.

2. Functional Requirements

a. What is the student going to see about the system ?

The student, when accesses his file, will see his matrix. This matrix is going to show the courses that are required for all students in that curriculum and provide blank spaces for the courses which can be track requirements, track electives or electives.

b. What is the curricular officer going to see about the system?

The curricular officer is the key person in the system. He is given the privileged access to all data in the system. He will completely oversee the process to make sure that it is proceeding properly and will be able to intervene to respond to any undesired events.

c. What functions are going to be performed by the students?

The student will be able to access only his portion of the database. Once logged in the system, he must ensure that he has entered the necessary information before the deadline. In other words, the system is going to be smart enough to warn the student when he logs in, about whether he has entered the required information at the specified time. If any of the courses that a student signed up for is not to be offered, then the student is required to make the appropriate modifications by consulting the curriculum officer, since he will not be allowed to make any kind of change for the following quarter after the specified deadline of the current quarter. The student will be able to view his schedule, verify the course selection in the following quarter, change the track of study and see any kind of information needed for track and course selections such as the lists of tracks, list of courses offered in the school, information about the default matrix, or information about the privileges given to him by the curriculum staff, whenever he needs it. If the student decides to drop or add a course, the necessary corrections will be performed by the curriculum staff. The student will be able to change his password for logging on to the system. He will also be able to read messages sent by the curriculum staff and send replies back to the curriculum officer.

d. What functions are going to be performed by the curriculum officer?

The curricular officer will oversee the process of course enrollment carried out by the students. He will be able to make any kind of modification to student information, the student matrix, course information or track information. He will input every new student in the database and store the desired default matrix in new student's record. He will do the necessary modifications to the matrices of the students, in the lists of the courses and the tracks when needed. He will also keep track of the attributes of courses and tracks such as the prerequisites for a course, the quarters that a course is planned to be offered, the list of courses required for a specific track, the list of courses as track electives for a specific track and the number of track requirements and so on. He will check the flags raised for those students who did not confirm their following quarter's course schedule yet. He will enter the system setting information such as the authorized password, the current quarter's start date, the number of hours that a student can enroll for in a quarter, the number of days allowed for the student's confirmation for the next quarter's schedule, the list of courses that could be duplicated and the list of the names of the default matrix types. He will also be able to assign privileges to the students such as the privilege for not being bound by the track requirements, default matrix or maximum hour limit. He will be able to enter courses that could be duplicated by a particular student and courses that a particular student has validated. He will enter or delete passwords for the students added to the database.

3. Constraints

a. Hardware Constraints:

There must be dedicated Personal Computers available for the students and curriculum staff.

b. Software Constraints:

The application program should be able to work on any "PC Windows" based machine with Microsoft Windows, Version 3.1 or higher as the operating environment.

c. Time Constraints :

The project must be completed by March 1995.

d. Reliability Constraints:

The software should be able to prevent the student from entering an invalid course or a course that is not going to be offered at that specific quarter.

The student will not be able to access other student's information.

The student will not be able to make changes in the current and previous quarters. For the following quarter, the students will be able to make changes until the deadline in the current quarter, designated by the curriculum officer.

The student will not be able to delete the core courses and enter courses different from the track electives that must be selected from.

If, at any given time, the matrix is not complying with the Department's requirements for graduation, then the student will be prompted to correct the problem. Whenever he attempts to make some changes for the following quarters, the program will check if the updated matrix is still complying with the graduation requirements.

With the current system, the student will not be allowed to take more than 6 courses or 20 hours per quarter. These restrictions on the numbers of the courses and course hours can be modified by the curriculum staff.

e. User Constraints:

The students in Computer Science Curriculum and the curriculum staff are the intended users of the target system. They are assumed to have known or have learned how to access to the portion of the data they need.

4. Responses To Undesired Events

If the user attempts to delete a core course, he will not be allowed to, and will be given an error message describing the problem.

If the student attempts to make changes to the past or current quarters, or to the following quarter after the deadline in the current quarter, he will not be allowed to do so and will be prompted by an error message.

If the student ends up with a matrix that falls short of the requirements of the graduation, he will be warned and will not be allowed to save the changes before correcting the problem. The system will also give options to the student about the courses among which he can choose, at any moment.

If the student fails to finish entering his courses by the due time, the curriculum staff will be able to identify and notify him to correct the discrepancy.

If the department cancels a course, those students who have signed up for the course should be able to make further changes under the supervision of the curriculum staff.

C. ESSENTIAL MODEL

1. Environment Model

a. Description of System Purpose

The new system is aimed at automating the course enrollment process in Computer Science Curriculum. In other words, the new system will make the process of keeping information about the students and courses and preparing an enrollment list to be sent to the department and the scheduler, more intuitive, more efficient, less time consuming and less error prone. Consequently the overall educational level of the students will be increased and the workload of the curricular staff will be reduced.

b. Event List

The event list is the list of the actions that a user starts in the system to perform some task. Following is the event list for the system to be implemented:

- Student views his current matrix.
- Student views his student information.
- Student changes his password.
- Student confirms next quarter.
- Student denies next quarter.
- Student enters a new course to the matrix

- Student changes a course in the matrix.
- Student enters a track.
- Student changes the track.
- Student gets information about valid courses.
- Student prints the course list.
- Student gets information about valid tracks.
- Student gets information about the default matrix.
- Student prints the default matrix.
- Student gets information about privileges given to him.
- Student checks the mail box.
- Student reads the messages received from the curriculum officer.
- Student sends message to the curriculum officer.
- Curriculum staff adds student to the database by entering student's information.
- Curriculum staff displays student's information.
- Curriculum staff updates student's information.
- Curriculum staff deletes student from the database.
- Curriculum staff displays the student list.
- Curriculum staff prints the student list.
- Curriculum staff adds course to the database by entering course's information.
- Curriculum staff displays course information.
- Curriculum staff updates course information.
- Curriculum staff deletes course from the database.
- Curriculum staff displays the course list.
- Curriculum staff prints the course list.
- Curriculum staff generates the enrollment list.
- Curriculum staff displays the enrollment list.
- Curriculum staff prints the enrollment list.
- Curriculum staff adds track with its track requirements to the database.

- Curriculum staff displays track information.
- Curriculum staff updates track and track requirements.
- Curriculum staff deletes track from the database.
- Curriculum staff adds password to the database.
- Curriculum staff displays password list.
- Curriculum staff deletes password from the database.
- Curriculum staff displays a student's matrix.
- Curriculum staff updates a student's matrix.
- Curriculum staff displays the privileges given to a student.
- Curriculum staff updates the privileges given to a student.
- Curriculum staff displays one of the default matrices.
- Curriculum staff updates one of the default matrices.
- Curriculum staff displays the system settings.
- Curriculum staff updates the system settings.
- Curriculum staff searches for a student from the database.
- Curriculum staff searches for a course from the database.
- Curriculum staff sends mail to a student or to all students.
- Curriculum staff displays the "Receive Log".
- Curriculum staff reads messages received from the students.
- Curriculum staff displays the "Send Log"
- Curriculum staff reads messages sent to students.
- Curriculum staff displays the list of students who did not confirm their next quarters' schedules.
- Curriculum staff prints the list of students who did not confirm their next quarters' schedules.
- Department works on the enrollment list.
- Department issues the list of courses.
- Department issues the list of courses to be offered for the following quarter.
- Scheduler receives the enrollment list and works on it.

- Scheduler issues the course schedule.
- Registrar gives academic information.
- Registrar collects academic information.

c. Context Diagram

The Context Diagram is the topmost diagram of the data flow diagrams in the design of the system and it shows the interaction of the outside users with the system. The context diagram of the Automated Course Enrollment System is shown in Figure 1.

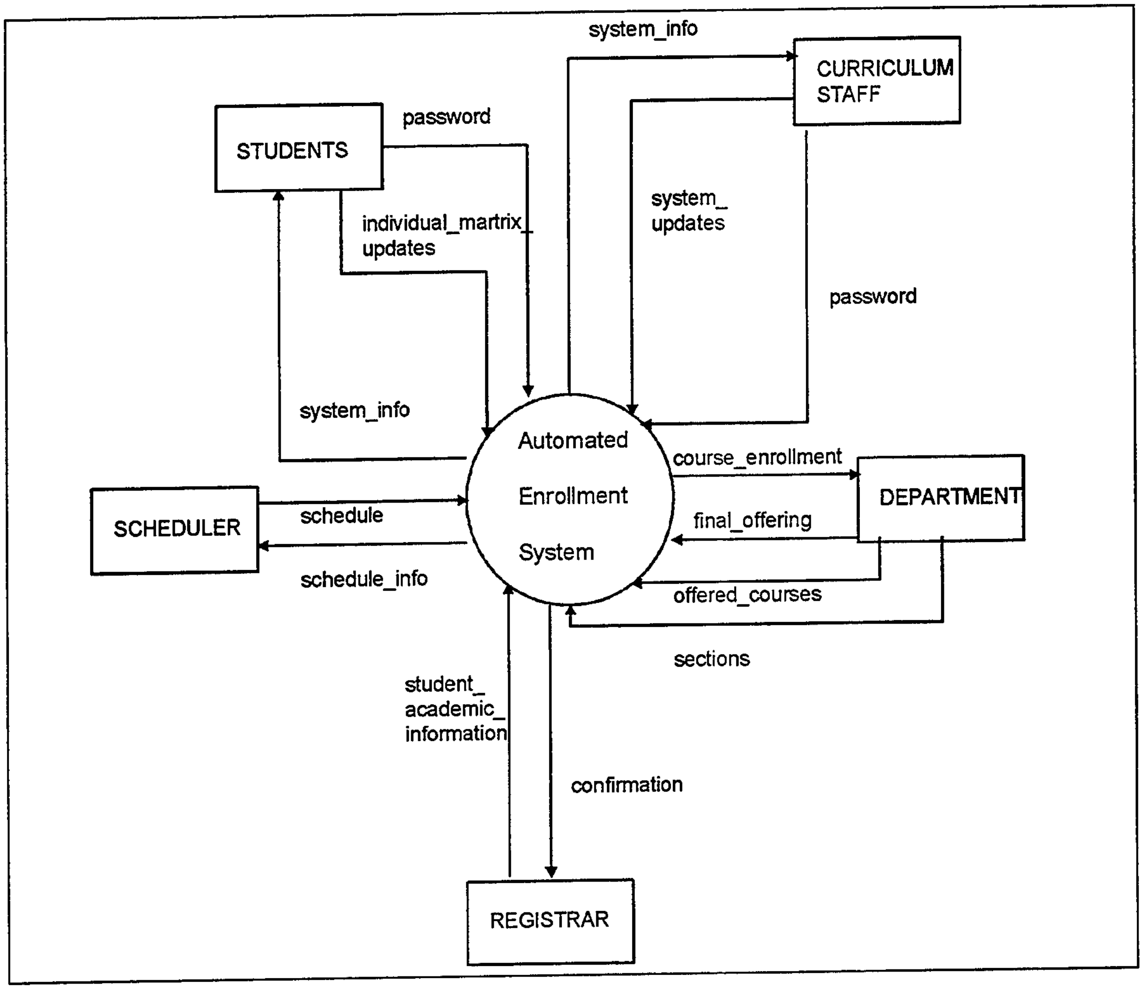


Figure 1. Context Diagram

2. Behavioral Model

The Behavioral Model reflects the internal behavior of the system and is comprised of Data Flow Diagrams, Entity-Relationship Diagram, Data Dictionary and Process Specifications.

a. Data Flow Diagrams

In the Automated Course Enrollment System, a large number of data flow diagrams has been developed. For that reason, only the data flow diagram in "Level 0" is shown in Figure 2. The other data flow diagrams are shown in Appendix A.

In the data flow diagram shown in Figure 2, there are five bubbles, each representing the actions taken by the five groups of users of the system, i.e., Students, Curriculum Staff, Scheduler, Department and Registrar.

b. Entity-Relationship Diagram

The Entity-Relationship Diagram is a diagram showing the relationships of the entities in a system. Here, it shows the relationships of the data entities processed by the system. The Entity-Relationship diagram for the Automated Course Enrollment System is shown in Figure 3.

c. Data Dictionary

The Data Dictionary is an alphabetical list of the data elements used in the data flow diagrams. This document serves the purpose of forming a common ground of understanding for the system analyst and the users. The Data Dictionary for Automated Course Enrollment System is shown in Appendix B.

d. Process Specifications

A Process Specification is a description of what is happening in a process within the data flow diagram. Here, a process specification for every bottom level process in the data flow diagrams is written. Due to its length, the process specifications for Automated Course Enrollment System are shown in Appendix C.

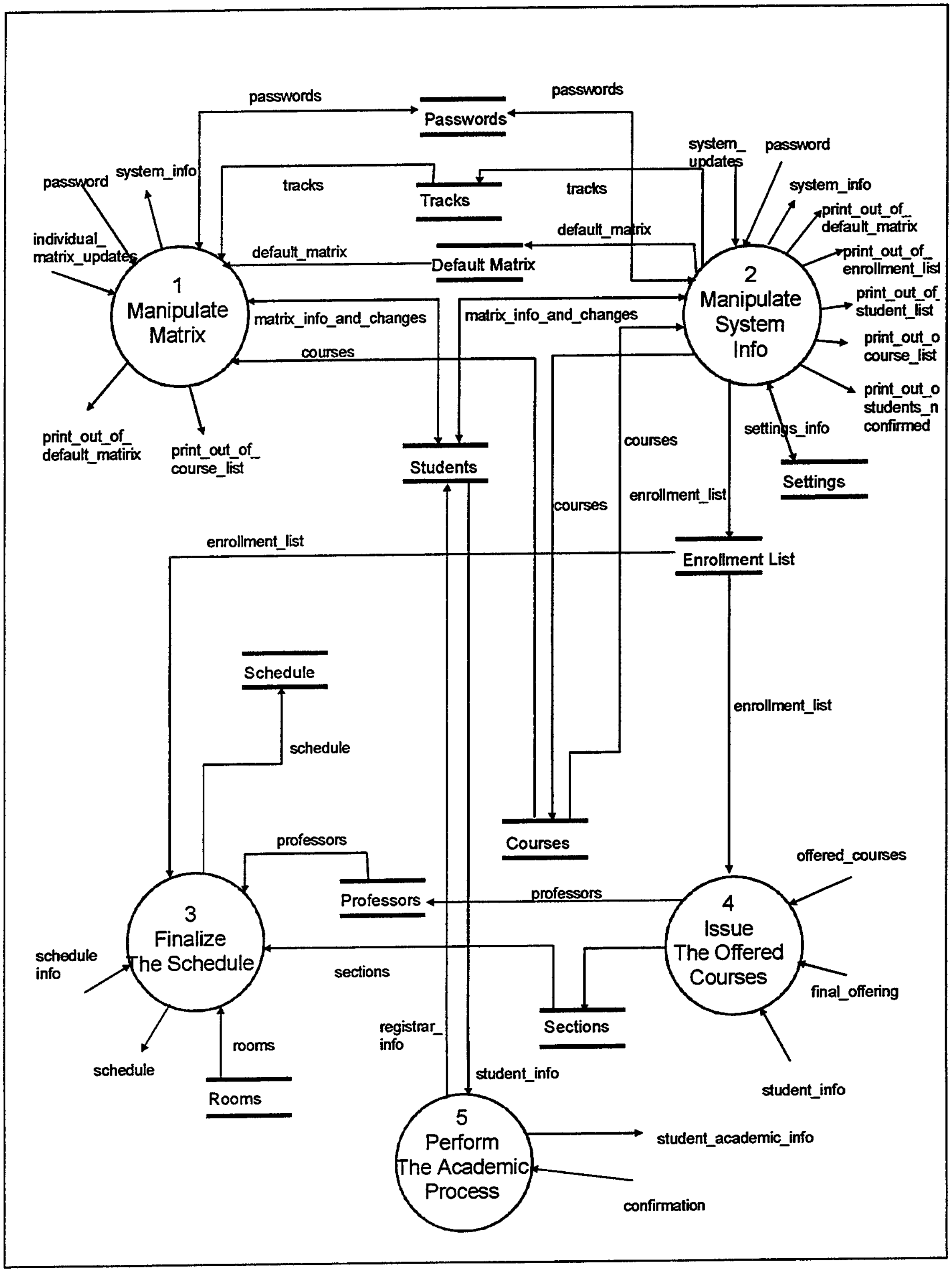


Figure 2. Data Flow Diagram Level 0

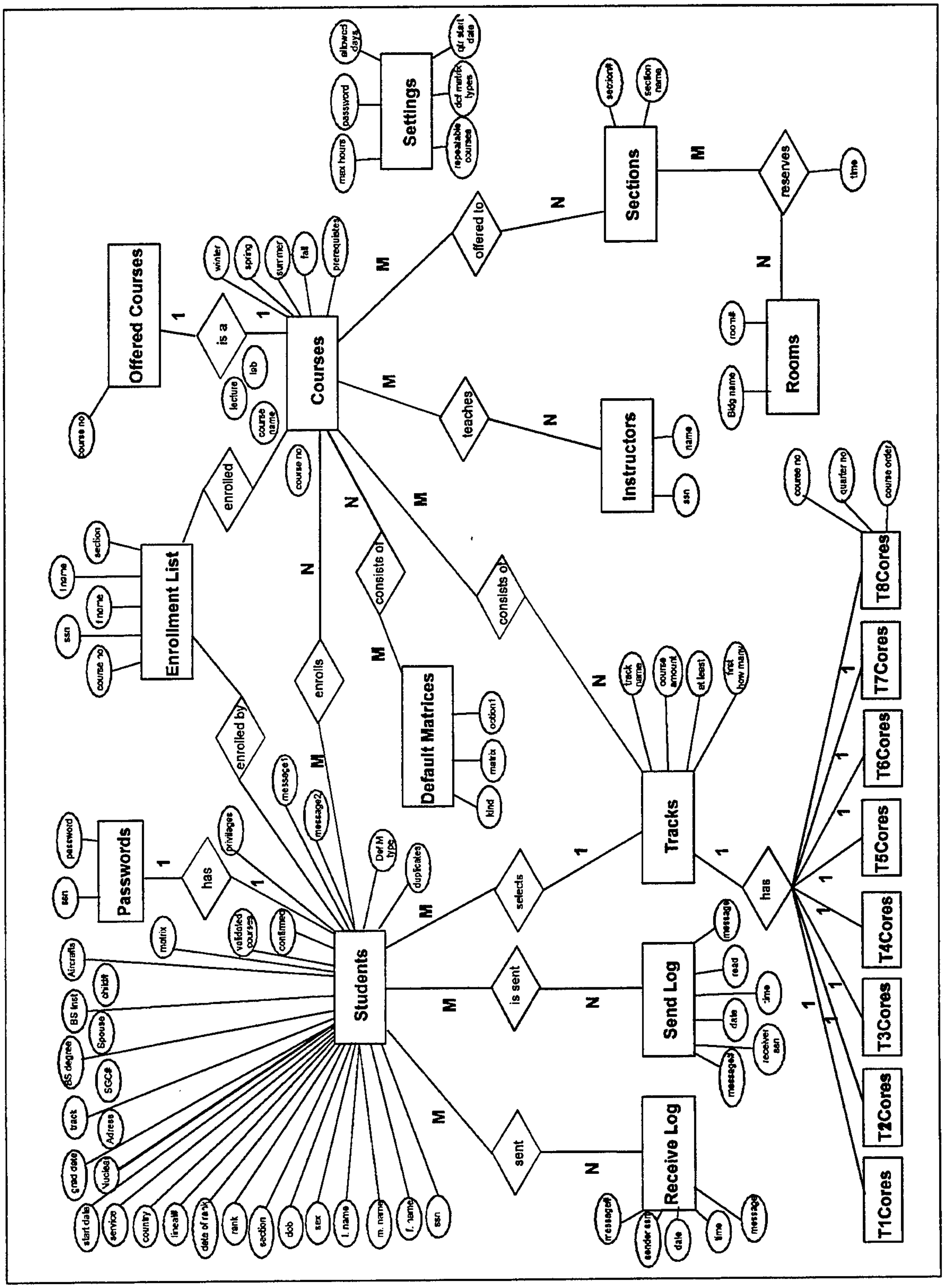


Figure 3. Entity-Relationship Diagram

D. USER IMPLEMENTATION MODEL

The User Implementation Model is where the detailed implementation issues are discussed.

1. Automation Boundary

The Automation Boundary is shown in Figure 4.

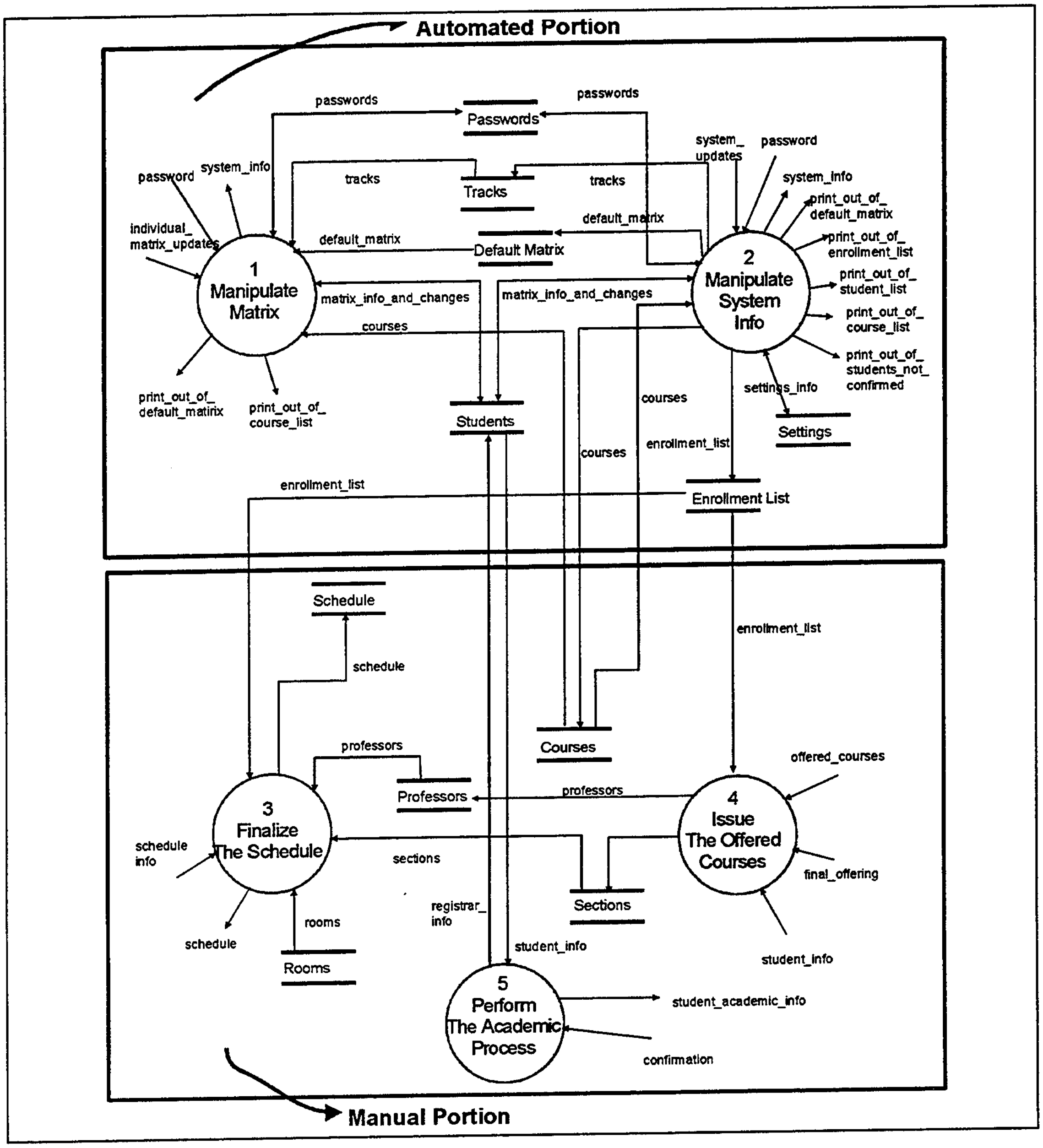


Figure 4. Automation Boundary

2. The Human Interface

a. Input Devices

- Conventional computer keyboard and mouse.
- Floppy Disk Drives.

b. Output Devices

- Any size of VGA Color Monitor with at least 800x 600 screen resolution.
- Laser or Dot Matrix Printer. (Laser printer is recommended)

c. Input/Ouput Formats

- The system will interact with the users by means of a GUI.
- The input process is facilitated by wide use of "dialog boxes" in a graphical user interface (GUI) environment.
- The user is prompted to enter the required data to the provided dialog boxes.
- There is no ordering of input in the same dialog box, but the user is prompted to input certain fields in order to finalize the input process successfully.
- During input of erroneous data, a warning feedback message will be provided to the user.
- Editing and error checking will be user dependent. In other words, there will be different procedures of error checking for students and curriculum staff.
- The user will be given the opportunity to cancel a job until a certain phase of data input is reached. This phase is usually going to be the moment after pressing the "OK" button.
- A convenient help mechanism for each operation will be made available to the user in all dialog boxes where there is an interaction between the user and the system.
- If any of the operations takes a while to finish, the user will be given a feedback to prevent him from thinking that the system has crashed.
- Defaults will be provided for standard inputs.
- The system will take advantage of color and sound without overdoing it.

3. Additional Manual Support Activities

Since there is a chance of having a hard drive failure or other system failure, the files have to be backed up to tapes and stored in convenient and secure locations.

4. Operational Constraints

- The room where the terminals and PCs will be kept should not be too humid, too cold or too hot so as to affect the electronic circuits in the hardware of the computers.
- Because of security considerations, the system will be run on dedicated PCs located in the curriculum room and it will not be connected to a Mainframe or Network.
- Since a GUI, based on a Windows system is targeted, Borland C++ is chosen as the programming language to write the software.

III. PROGRAMMING CONSIDERATIONS

A. DEVELOPMENT HARDWARE UTILIZED

A Personal Computer with the following specifications:

- Pentium-60 MHz processor
- 16 MB RAM
- 424 MB Hard Disk
- 2 MB VRAM

B. DEVELOPMENT SOFTWARE UTILIZED

1. Operating System

Windows 3.11 for Workgroups.

2. Programming Language

Borland C++ 4.0.

3. Development Tools

a. Resource Workshop Version 4.0

This is a utility that comes with the Borland C++ 4.0 Application. It is used to create and edit resource files. These resource files can include icons, bitmaps, cursors, fonts, dialog boxes, menus, accelerators and string tables.

In the "Automated Course Enrollment System" application, the Resource Workshop was used to a large extent to create and edit dialog boxes, icons, bitmaps, menus and accelerators used in the application.

b. Paradox Database Engine

This is a utility that does not come with Borland C++ Application and it has to be obtained separately. It can be used to give a Borland C++ Application the capability to create, access and edit "Paradox Database Files" within a C++ program. The nice thing about Paradox Database Engine is that, if the database files, where the application related information is being kept, are being used by multiple users concurrently, it provides means to preserve the data integrity. In the "Automated Course Enrollment System" application, all of the application related information such as information about Students, Courses, Tracks etc., was stored in Paradox Database Tables.

c. Crystal Reports Version 1.1

This is a utility that can be used to create print forms by using the data from the Paradox Database Tables. In the "Automated Course Enrollment System" application, the print forms such as Student List, Course List, Enrollment List etc., were created by using "Crystal Reports" application.

d. Help Compiler

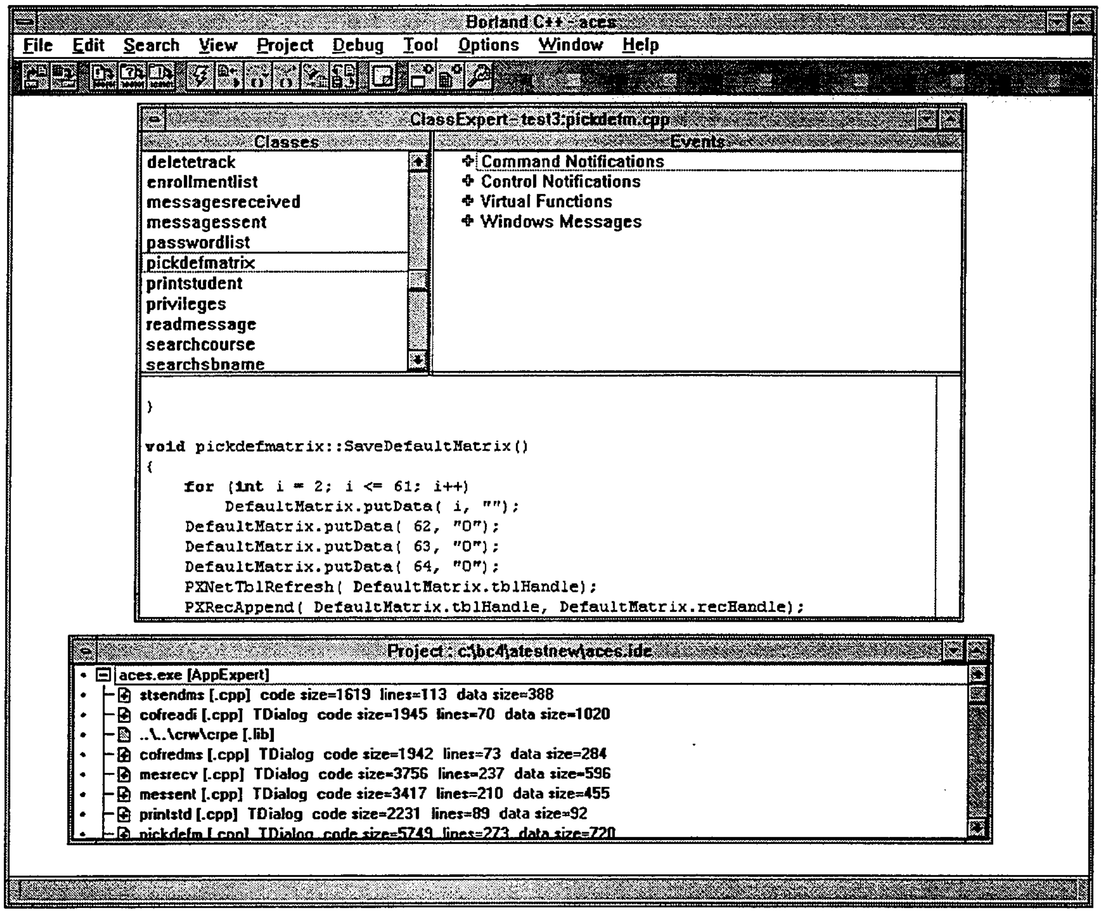
This is a utility that comes with the Borland C++ Application. It is used to compile a file with ".hpj" extension to create a help file under the same name with ".hlp" extension. Then the file with ".hlp" extension can be executed by Windows help file execution program.

C. HOW IS THE SOFTWARE USED?

1. Borland C++ 4.0

Since Borland C++ 4.0 is a Windows Application, its graphical user interface has made the job of writing the code a lot easier. This version of Borland C++ has come with some utilities which makes "Windows Application Development" a lot easier. In Figure 5, we can see the general look of application window.

Borland C++ 4.0 Application has some tools integrated into the application window. These tools are aimed at making the process of application development easier and farther fromt the lower level programming issues.



```
}

void pickdefmatrix::SaveDefaultMatrix()
{
    for (int i = 2; i <= 61; i++)
        DefaultMatrix.putData( i, "");
    DefaultMatrix.putData( 62, "0");
    DefaultMatrix.putData( 63, "0");
    DefaultMatrix.putData( 64, "0");
    PXNetTblRefresh( DefaultMatrix.tblHandle);
    PXRecAppend( DefaultMatrix.tblHandle, DefaultMatrix.recHandle);
```

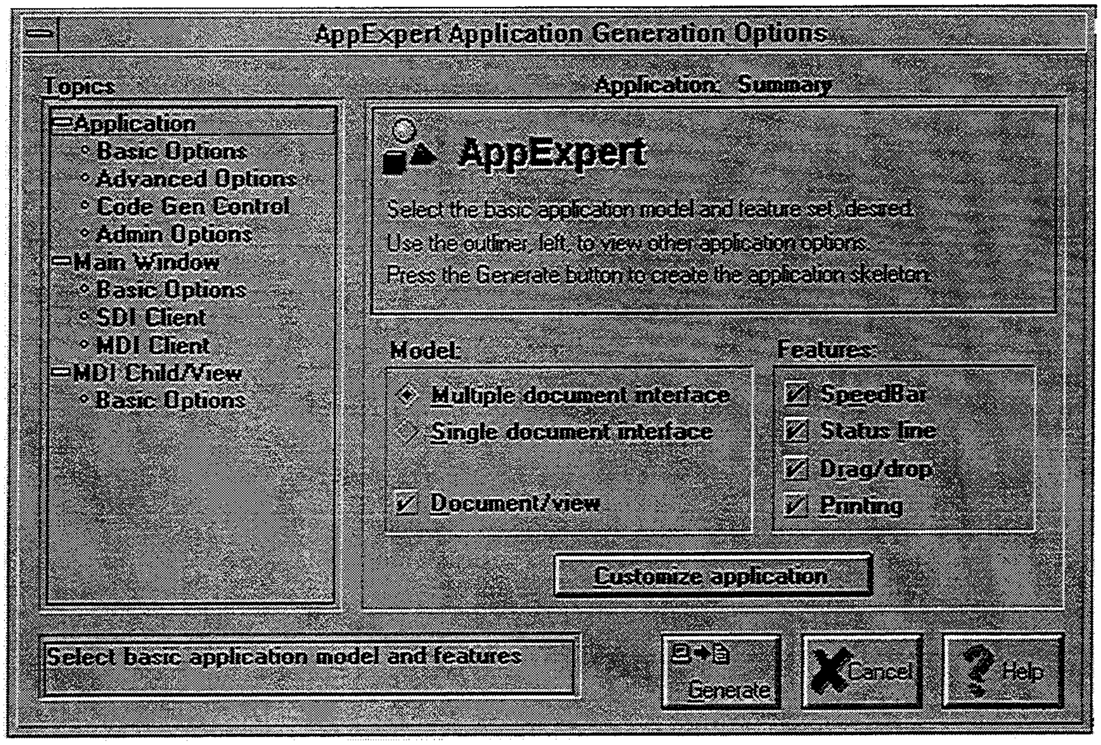
```
aces.exe [AppExpert]
  stsendms [.cpp]  code size=1619  lines=113  data size=388
  cofreadi [.cpp]  TDialog  code size=1945  lines=70  data size=1020
  ..\..\crw\crpe [.lib]
  cofredms [.cpp]  TDialog  code size=1942  lines=73  data size=284
  mesrecv [.cpp]  TDialog  code size=3756  lines=237  data size=596
  messent [.cpp]  TDialog  code size=3417  lines=210  data size=455
  printstd [.cpp]  TDialog  code size=2231  lines=89  data size=92
  pickdefm [.cpp]  TDialog  code size=5749  lines=273  data size=720
```

Figure 5. Borland C++ 4.0 Application Desktop

a. Application Expert

Borland C++ User's Guide[Ref. 1] explains "Application Expert" as a tool to create a Windows executable with features such as a "speed bar", a "status bar", a "menu structure", an "online help", and a "MDI(Multiple Document Interface) windows".

In other words, the Application Expert, or AppExpert for short, can be used to create a basis for the windows application. By selecting "AppExpert..." command from the "Project" menu item in the application menu bar, a programmer can enter the name and the other options to create a basic windows application, on which he can later add the functionality specific to the application. In Figure 6, we see the dialog box that appears after a name for the application is entered.



Select the basic application model and feature set desired.
Use the outliner, left, to view other application options.
Press the Generate button to create the application skeleton.

Model:

◆ Multiple document interface
◇ Single document interface
☑ Document/view

Features:

☑ SpeedBar
☑ Status line
☑ Drag/drop
☑ Printing

Select basic application model and features

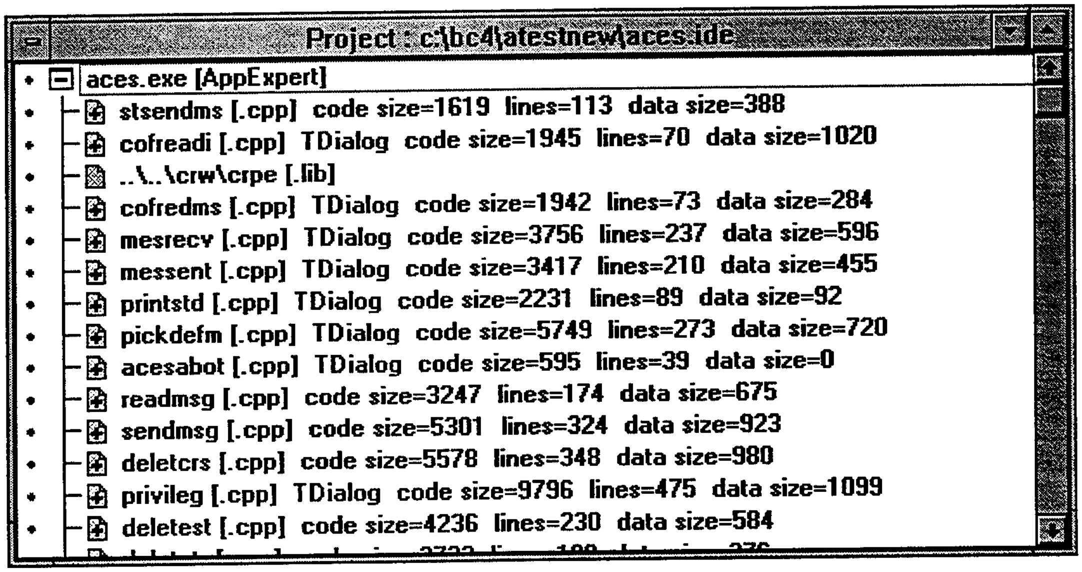
Figure 6. Application Expert Application Generation Dialog Box

In the dialog box called " AppExpert Application Generation Options", the user is prompted to enter the options for "Application", "Main Window" and "MDI Child/View" before he can press the "Generate" button to generate the code for the basic application. After the code is generated, the user can go to the IDE(Integrated Development Environment) as shown in Figure 5, to give the application the shape that he would like.

b. Project Manager

Borland C++ User's Guide[Ref. 2] explains "Project Manager" as the tool to handle the applications that are built from many components. It organizes and updates complex applications by keeping track of all the files and their interdependencies in a project file with the extension ".IDE".

The "Project Manager" makes the process of creating an executable file with an ".EXE" extension a lot easier, because it checks the interdependencies between the components of the project and compiles the files that changed since the last build of the project. The "Project Manager" is displayed as a window in which the files of the project are shown hierarchically. Figure 7 shows the project manager of "Automated Course Enrollment System".

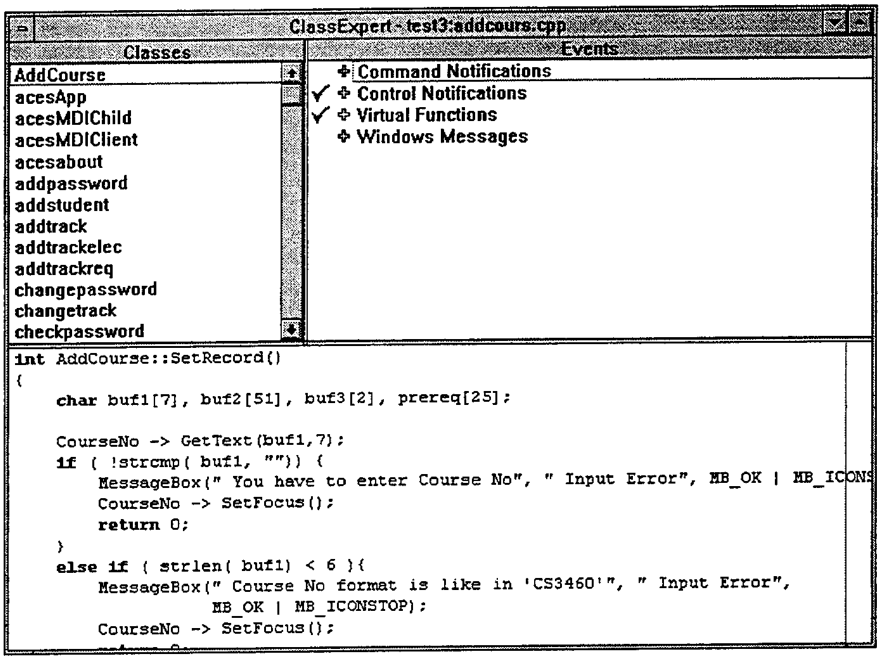


```
Project : c:\bc4\atestnew\aces.ide
• ⊟ aces.exe [AppExpert]
•   ├─ stsendms [.cpp]  code size=1619  lines=113  data size=388
•   ├─ cofreadi [.cpp]  TDialog  code size=1945  lines=70  data size=1020
•   ├─ ..\..\crw\crpe [.lib]
•   ├─ cofredms [.cpp]  TDialog  code size=1942  lines=73  data size=284
•   ├─ mesrecv [.cpp]  TDialog  code size=3756  lines=237  data size=596
•   ├─ messent [.cpp]  TDialog  code size=3417  lines=210  data size=455
•   ├─ printstd [.cpp]  TDialog  code size=2231  lines=89  data size=92
•   ├─ pickdefm [.cpp]  TDialog  code size=5749  lines=273  data size=720
•   ├─ acesabot [.cpp]  TDialog  code size=595  lines=39  data size=0
•   ├─ readmsg [.cpp]  code size=3247  lines=174  data size=675
•   ├─ sendmsg [.cpp]  code size=5301  lines=324  data size=923
•   ├─ deletcrs [.cpp]  code size=5578  lines=348  data size=980
•   ├─ privileg [.cpp]  TDialog  code size=9796  lines=475  data size=1099
•   ├─ deletest [.cpp]  code size=4236  lines=230  data size=584
```

Figure 7. Project Manger Window

c. Class Expert

Borland C++ User's Guide[Ref. 2] explains "Class Expert" as the tool to create new classes, edit and refine the implementation of classes, and navigate through the source code for existing classes in "Application Expert" applications. Figure 8 shows the "Class Expert" for "Automated Course Enrollment System".



```
int AddCourse::SetRecord()
{
    char buf1[7], buf2[51], buf3[2], prereq[25];

    CourseNo -> GetText(buf1,7);
    if ( !strcmp( buf1, "")) {
        MessageBox(" You have to enter Course No", " Input Error", MB_OK | MB_ICONS
        CourseNo -> SetFocus();
        return 0;
    }
    else if ( strlen( buf1) < 6 ){
        MessageBox(" Course No format is like in 'CS3460'", " Input Error",
                   MB_OK | MB_ICONSTOP);
        CourseNo -> SetFocus();
```

Figure 8. Class Expert Window

In "Class Expert" dialog box, the programmer can also manipulate the events for the class highlighted in the adjacent section called "Classes". The simplest example of an event can be described as an action that should to take place when a user presses a push button in a dialog box.

2. Resource Workshop

As described earlier, the "Resource Workshop" is a utility that can be used to create and edit resource files that can include icons, bitmaps, cursors, fonts, dialog boxes, menus, accelerators and string tables. In Figure 9, the resource file for "Automated Course Enrollment System" is shown in "Resource Workshop" application window.

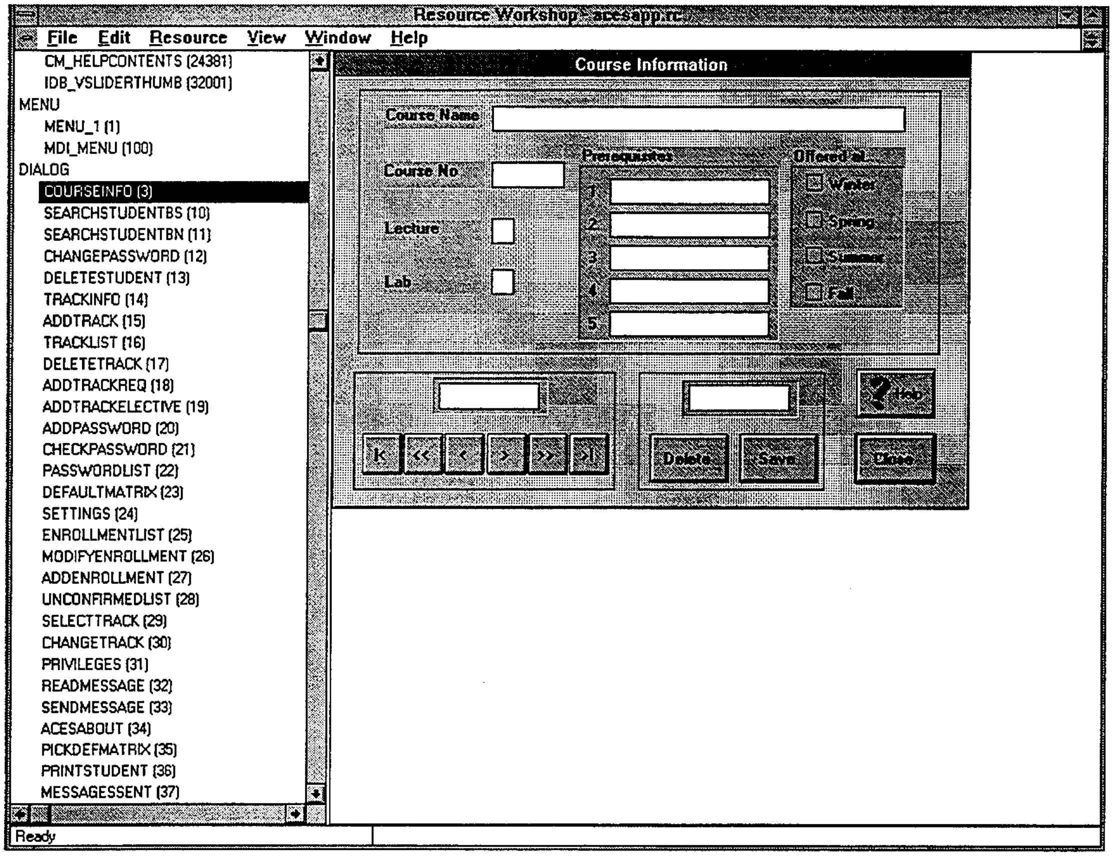


Figure 9. Resource Workshop

a. Bitmap Editor

In the "Bitmap Editor", the programmer can create or edit any bitmap resource, including "icons", "cursors", "bitmaps" and "fonts". In the ACES application, the "Bitmap Editor" is used to create the application icons and the bitmaps used for the quick access buttons in the toolbar of the application window. In Figure 10, the "Bitmap Editor" is shown with one of the toolbar bitmaps opened for editing.

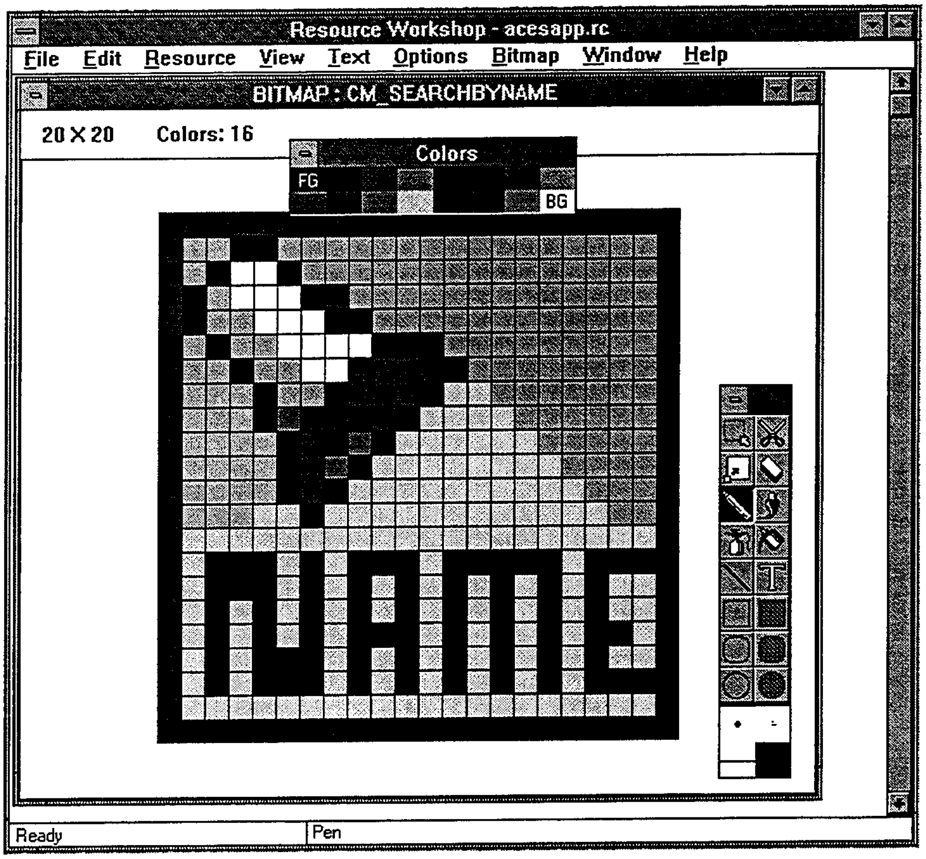

Resource Workshop - acesapp.rc
File Edit Resource View Text Options Bitmap Window Help
BITMAP : CM_SEARCHBYNAME
20 X 20
Colors: 16
Colors
FG
BG
Ready
Pen

Figure 10. Bitmap Editor

b. Dialog Editor

In the "Dialog Editor", the programmer can create and edit "Dialog Boxes" to be used in the application. This utility provides tools to create and edit "Dialog Boxes" with mouse drag and drop support. Its highly developed user-interface makes the job of creating nice looking dialog boxes a very quick process. In the ACES application, the "Dialog Editor" was used to create almost 40 dialog boxes.

In Figure 11, the "Dialog Editor" is shown with one of the dialog boxes used in the ACES application, ready to be edited.

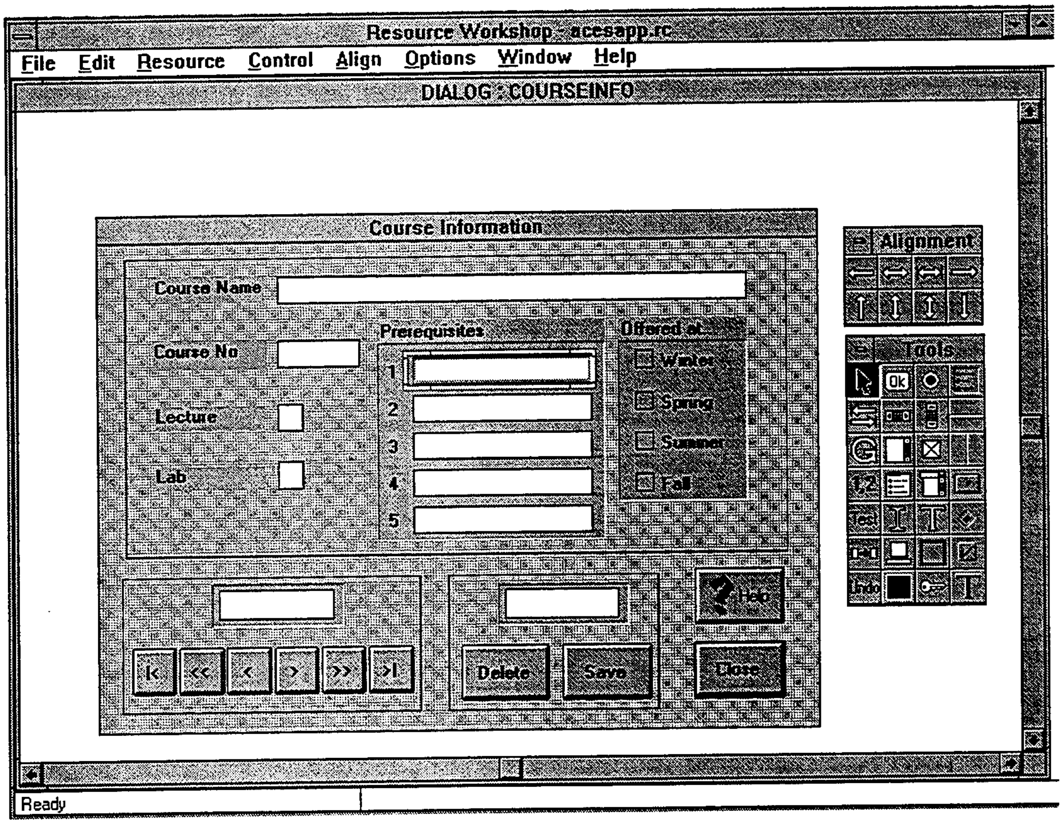
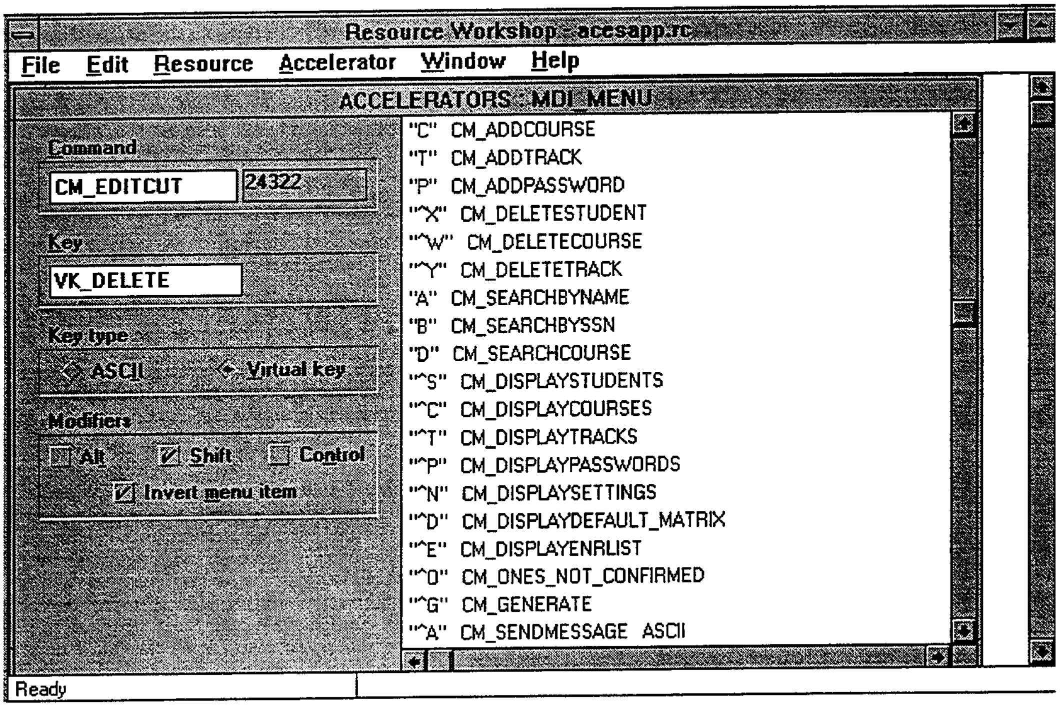


Figure 11. Dialog Editor

c. Accelerator Editor

In the "Accelerator Editor", the programmer can create and edit "accelerators" for his application. An accelerator can be defined as a key combination the user presses to perform a task with an application. In the ACES application, the "Accelerator Editor" is used to create "accelerators" for the menu commands in the menu bar. In Figure 12, the "Accelerator Editor" is shown with one of the menu bars used in theACES application.



Command: CM_EDITCUT 24322

Key: VK_DELETE

Key type: ○ ASCII ● Virtual key

Modifiers: ☐ Alt ☑ Shift ☐ Control

☑ Invert menu item

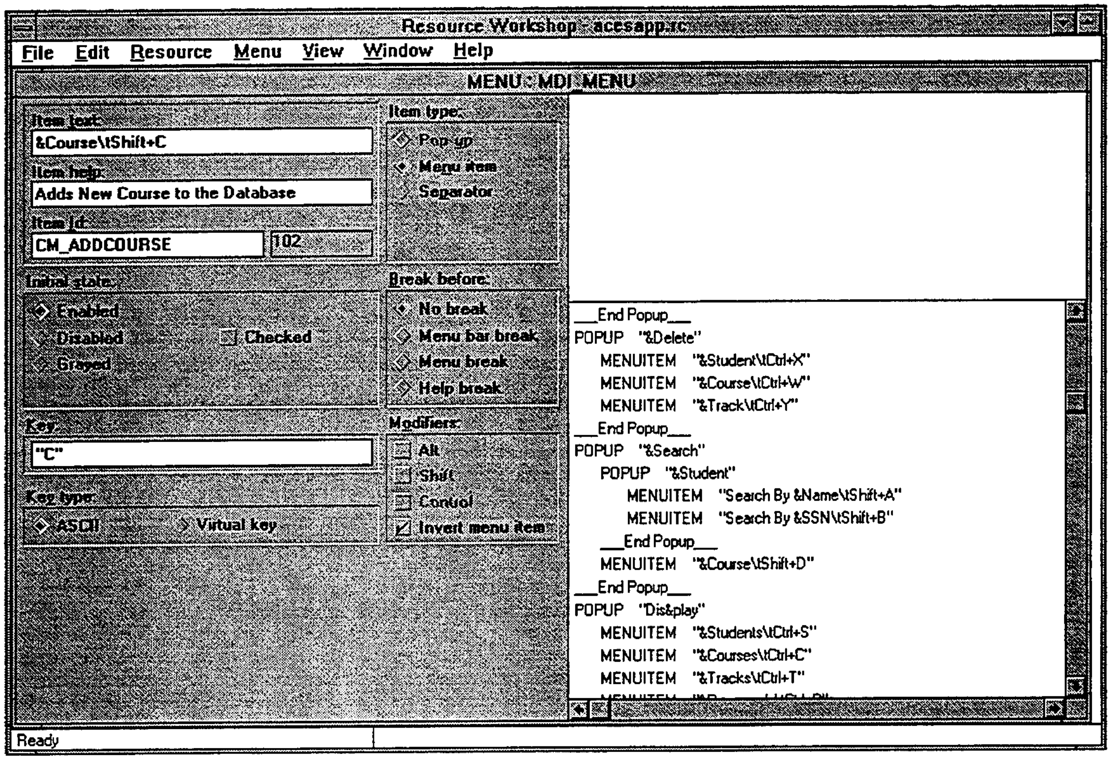
```
"C"  CM_ADDCOURSE
"T"  CM_ADDTRACK
"P"  CM_ADDPASSWORD
"^X"  CM_DELETESTUDENT
"^W"  CM_DELETECOURSE
"^Y"  CM_DELETETRACK
"A"  CM_SEARCHBYNAME
"B"  CM_SEARCHBYSSN
"D"  CM_SEARCHCOURSE
"^S"  CM_DISPLAYSTUDENTS
"^C"  CM_DISPLAYCOURSES
"^T"  CM_DISPLAYTRACKS
"^P"  CM_DISPLAYPASSWORDS
"^N"  CM_DISPLAYSETTINGS
"^D"  CM_DISPLAYDEFAULT_MATRIX
"^E"  CM_DISPLAYENRLIST
"^O"  CM_ONES_NOT_CONFIRMED
"^G"  CM_GENERATE
"^A"  CM_SENDMESSAGE  ASCII
```

Ready

Figure 12. Accelerator Editor

d. Menu Editor

In the "Menu Editor", the programmer can create and edit "menus" for Windows applications. In the ACES application, the "Menu Editor" was used to create two menus to be used in two different modes of the application. In Figure 13, the "Menu Editor" is shown with one of the two menus used in the ACES application, ready to be edited.


Resource Workshop - acesapp.rc
File Edit Resource Menu View Window Help
MENU : MDI_MENU
Ready

Item text: &Course\tShift+C

Item help: Adds New Course to the Database

Item id: CM_ADDCOURSE 102

Item type: ◇ Pop-up ◆ Menu item ◇ Separator

Initial state: ◆ Enabled ◇ Disabled ◇ Grayed □ Checked

Break before: ◆ No break ◇ Menu bar break ◇ Menu break ◇ Help break

Key: "C"

Key type: ◆ ASCII ◇ Virtual key

Modifiers: □ Alt □ Shift □ Control ☑ Invert menu item

```
__End Popup__
POPUP   "&Delete"
   MENUITEM   "&Student\tCtrl+X"
   MENUITEM   "&Course\tCtrl+W"
   MENUITEM   "&Track\tCtrl+Y"
__End Popup__
POPUP   "&Search"
   POPUP   "&Student"
      MENUITEM   "Search By &Name\tShift+A"
      MENUITEM   "Search By &SSN\tShift+B"
   __End Popup__
   MENUITEM   "&Course\tShift+D"
__End Popup__
POPUP   "Dis&play"
   MENUITEM   "&Students\tCtrl+S"
   MENUITEM   "&Courses\tCtrl+C"
   MENUITEM   "&Tracks\tCtrl+T"
```

Figure 13. Menu Editor

3. Paradox Database Engine

The Paradox Database Engine provides C++ and some other programming languages with the capability for making use of the power of the "Paradox" application. It has over 90 functions that can be used within the C++ program once the header file for the database engine is included in the file where the references to the database engine functions are to be made. There is no application window to be opened for "Paradox Database Engine", but there is a configuration utility which opens a window for the user to enter resource limits, such as the maximum number of tables and files that can be opened at the same time. This window also provides a network configuration utility, if the application is to be run in a network. Figure 14 displays this configuration utility window.

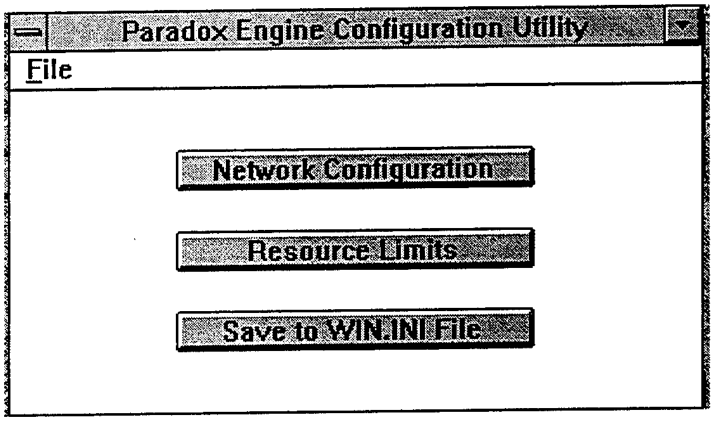
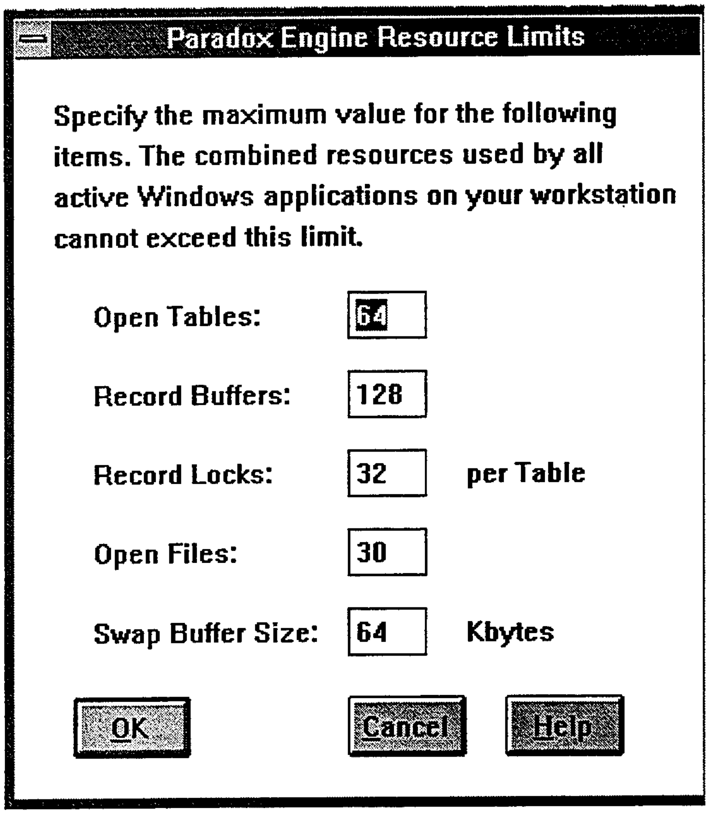


Figure 14. Paradox Database Engine Configuration Utility

In Figure 15, the Resource Limits Dialog Box is shown. This dialog box has been used several times in the course of development of the ACES application to make changes to the resource limits due to changing resource requirements.

In the ACES application, a class named "Table" was created to make use of the database engine. This "Table" class encompasses the database engine functions which enable the programmer to access the paradox tables and manipulate them. More detailed information as to how this is accomplished will be given in later sections when the files used in the program are being closely examined.


Paradox Engine Resource Limits

Specify the maximum value for the following items. The combined resources used by all active Windows applications on your workstation cannot exceed this limit.

Open Tables: 64

Record Buffers: 128

Record Locks: 32 per Table

Open Files: 30

Swap Buffer Size: 64 Kbytes

Figure 15. Paradox Database Engine Resource Limits Dialog Box

4. Crystal Reports

Crystal Reports is a utility to create print forms separately based on the records of the paradox database tables. The nice thing about Crystal Reports is that the programmer can create the print forms in a very user-friendly and powerful environment and later can make calls to these print forms from within the C++ code by including the header file for Crystal reports to the code.

In Figure 16, we can see the general look of the Crystal Reports application window with one of the print reports created for the ACES application.

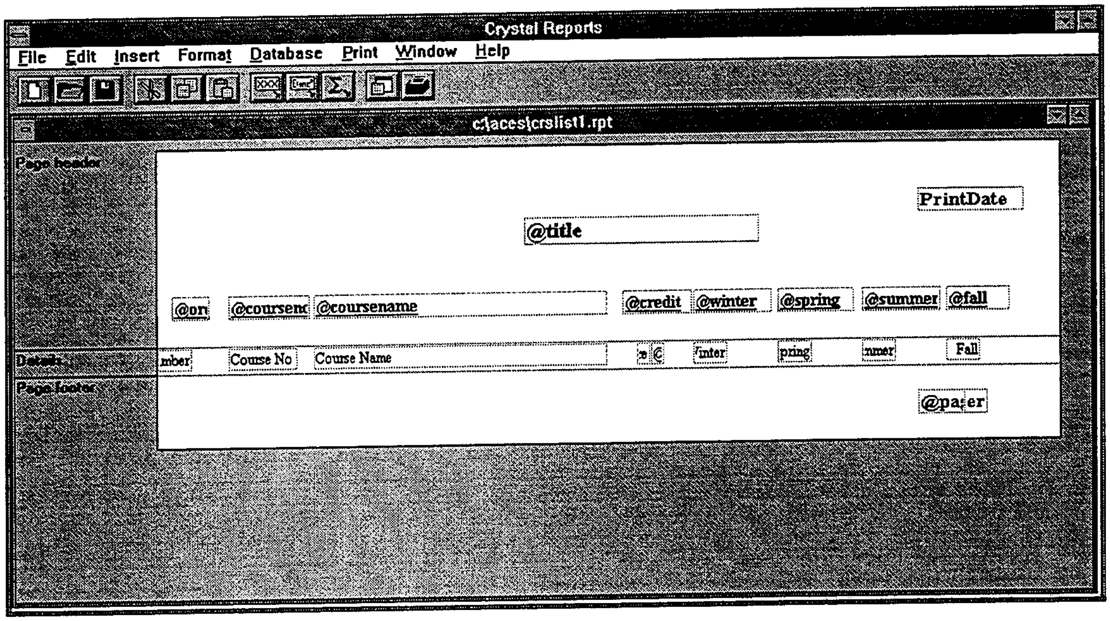
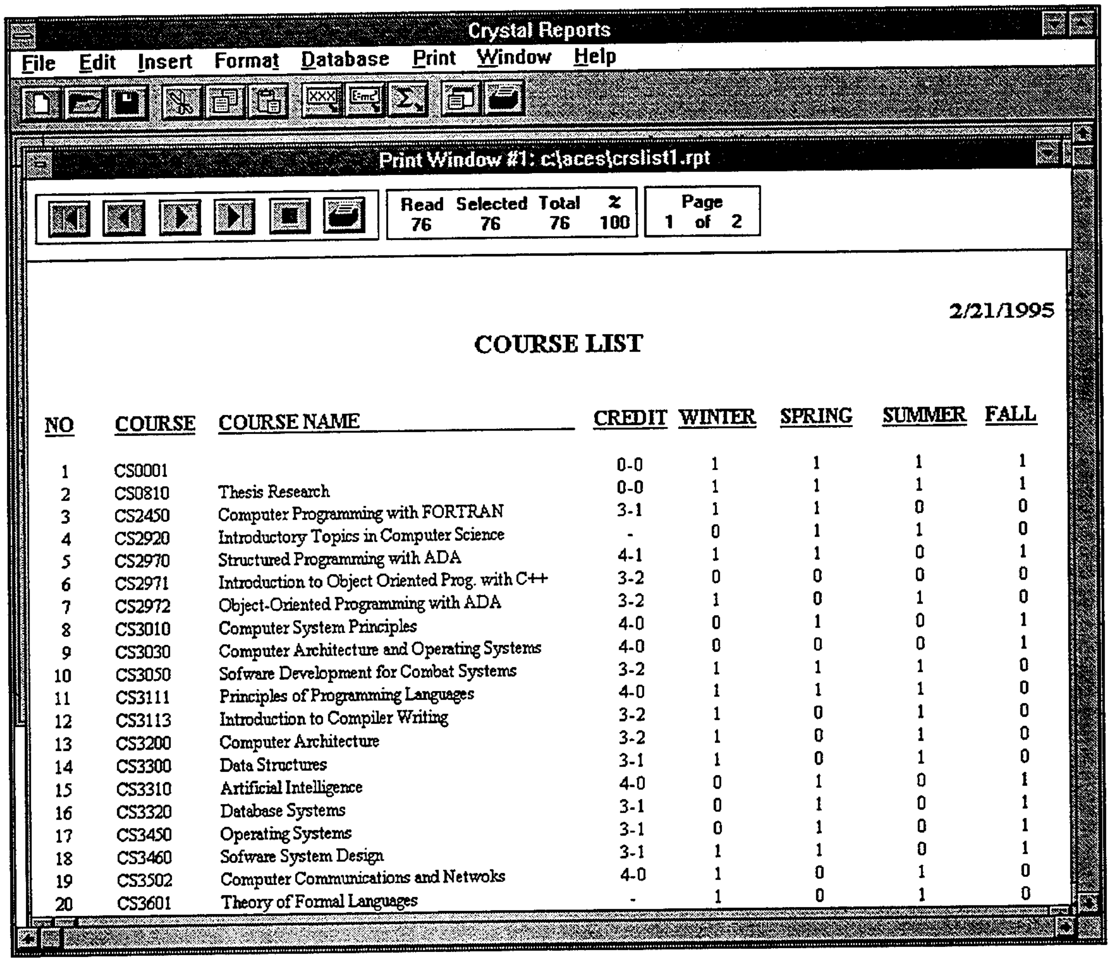


Figure 16. Crystal Reports Application Window

The users of Crystal Reports are not restricted to use the utility within an application. It can also be used directly to create print forms from paradox tables and print them. But the real power of Crystal Reports can be observed when it is incorporated into an application program like ACES application.

In Figure 17, the example print out of the print form displayed in Figure 16 is shown.


Crystal Reports
File Edit Insert Format Database Print Window Help
Print Window #1: c:\aces\crslist1.rpt
Read 76
Selected 76
Total 76
% 100
Page 1 of 2

2/21/1995

COURSE LIST

NO	COURSE	COURSE NAME	CREDIT	WINTER	SPRING	SUMMER	FALL
1	CS0001		0-0	1	1	1	1
2	CS0810	Thesis Research	0-0	1	1	1	1
3	CS2450	Computer Programming with FORTRAN	3-1	1	1	0	0
4	CS2920	Introductory Topics in Computer Science	-	0	1	1	0
5	CS2970	Structured Programming with ADA	4-1	1	1	0	1
6	CS2971	Introduction to Object Oriented Prog. with C++	3-2	0	0	0	0
7	CS2972	Object-Oriented Programming with ADA	3-2	1	0	1	0
8	CS3010	Computer System Principles	4-0	0	1	0	1
9	CS3030	Computer Architecture and Operating Systems	4-0	0	0	0	1
10	CS3050	Sofware Development for Combat Systems	3-2	1	1	1	0
11	CS3111	Principles of Programming Languages	4-0	1	1	1	0
12	CS3113	Introduction to Compiler Writing	3-2	1	0	1	0
13	CS3200	Computer Architecture	3-2	1	0	1	0
14	CS3300	Data Structures	3-1	1	0	1	0
15	CS3310	Artificial Intelligence	4-0	0	1	0	1
16	CS3320	Database Systems	3-1	0	1	0	1
17	CS3450	Operating Systems	3-1	0	1	0	1
18	CS3460	Sofware System Design	3-1	1	1	0	1
19	CS3502	Computer Communications and Netwoks	4-0	1	0	1	0
20	CS3601	Theory of Formal Languages	-	1	0	1	0

Figure 17. Sample Print Out of the Print Report Shown in Figure 16.

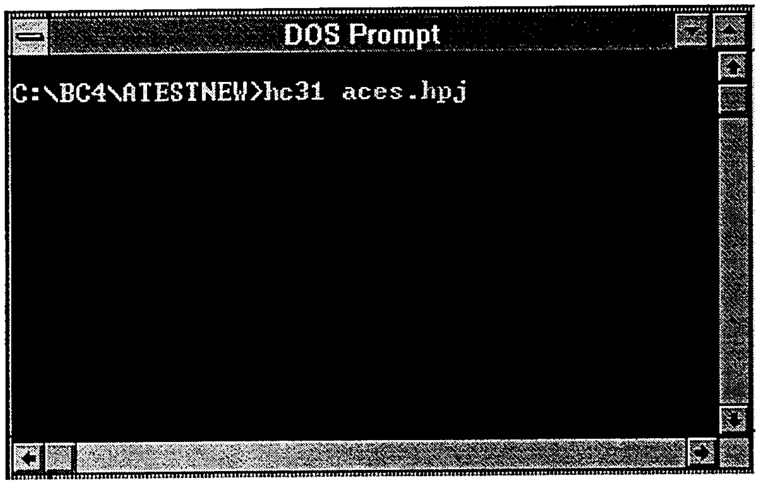
5. Help File Compiler

The Help Compiler is a utility that comes with the Windows 3.x operating system and it an executable under the name "hc31.exe". This executable file compiles a file with ".hpj" extension, to create a help file with a ".hlp" extension that can be executed by using "Winhelp" executable. In a ".hpj" file, the "hc31.exe" compiler gets the names of the ".rtf" files which will be used as the resource for the help file to be created. The ".hpj" file in the ACES application named "aces.hpj" is shown in Figure 18. As can be seen in Figure 18, the "aces.hpj" file refers to some files with ".rtf" extensions. These files with ".rtf" extensions are called files with "Rich Text Format". In other words, the help file compiler can interpret the files with "Rich Text Format" created by any word processor that supports "Rich Text Format". "Wordperfect 5.2" was used to write four RTF files in "aces.hpj" file. While creating the basic application with "Application Expert" in Borland C++, some of these files are automatically generated, if the help file option is checked before the Application Expert was prompted to generate the code.

```
[OPTIONS]
TITLE=ACES Help
CONTENTS=main_index
COMPRESS=true
WARNING=2
ERRORLOG = errorlog.txt

[FILES]
mainhelp.rtf     ; main topics
dlgboxes.rtf     ; dialog boxes
toolbar.rtf      ; toolbar topics
keyboard.rtf     ; keyboard topics

[BITMAPS]
helpicon.bmp

[MAP]
#include <acesapp.rh>
```

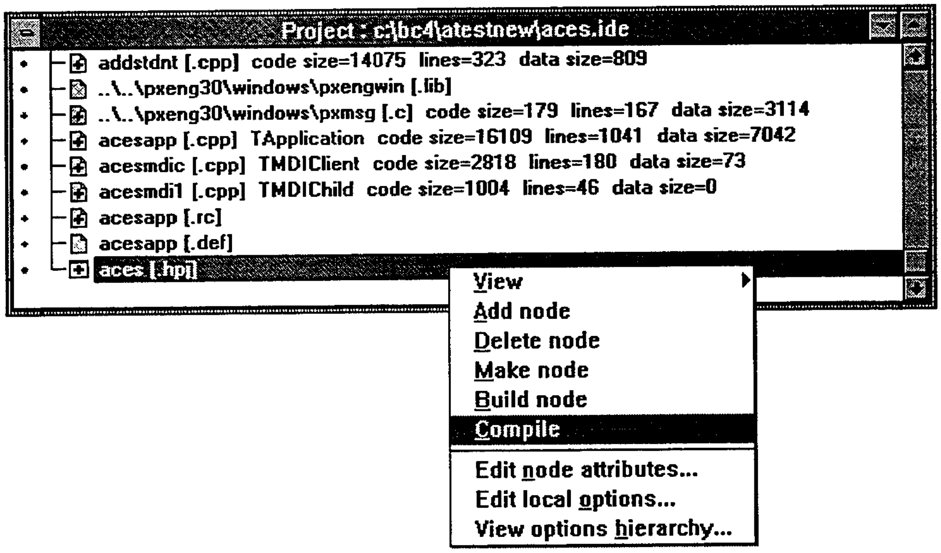
Figure 18. "aces.hpj" File Compiled to Get "aces.hlp" File in ACES Application

The "hc31.exe" help file compiler can be used directly from the "DOS" prompt as shown in Figure 19, or it can be executed from the project manager window in Borland C++ application window if the ".hpj" file is included in the project as a node. This is shown in Figure 20.



```
C:\BC4\ATESTNEW>hc31 aces.hpj
```

Figure 19.Help File Generation at DOS Command Prompt



```
addstdnt [.cpp]  code size=14075  lines=323  data size=809
..\..\pxeng30\windows\pxengwin [.lib]
..\..\pxeng30\windows\pxmsg [.c]  code size=179  lines=167  data size=3114
acesapp [.cpp]  TApplication  code size=16109  lines=1041  data size=7042
acesmdic [.cpp]  TMDIClient  code size=2818  lines=180  data size=73
acesmdi1 [.cpp]  TMDIChild  code size=1004  lines=46  data size=0
acesapp [.rc]
acesapp [.def]
aces [.hpj]
```

Figure 20. Running "hc31.exe" Help File Compiler from within Borland C++

The created ".hlp" files can be run from within the C++ code or directly from the Windows Program Manager by selecting "File/New" menu option to create a new program item.

The information regarding how to write ".rtf" files in a word processor is widely explained in the online help manual that came with Borland C++ under the name "cwh.hlp".

D. PROGRAM STRUCTURE

1. Class Hierarchy

ACES application has been written with an object-oriented approach. Borland C++ 4.0 has provided a large library of classes and functions that any programmer can utilize in a program. A special library called "Object Windows Library" or "OWL" for short, has made writing Windows applications easier with its classes which can be used to create a "Graphical User Interface (GUI)". Class derivation is the heart of using OWL classes. Normally, the OWL programmers do not directly use the classes provided in OWL. Instead, the user creates his own class that is derived from one of the existing OWL classes in order to provide additional functionality.

Figure 21 presents the class hierarchy of the ACES application.

2. Files

a. Main File

This main source file is named "acesApp.cpp". This file is the source file for the header file "acesApp.h" in which the class "acesApp" is derived from the OWL class "TApplication". In this file, the following tasks are accomplished:

- A "Log On Dialog Box" is displayed and the SSN# and the password of the user is prompted.
- If a valid SSN# and corresponding password are entered, it is investigated whether the user who is logging on is a curriculum staff or not.

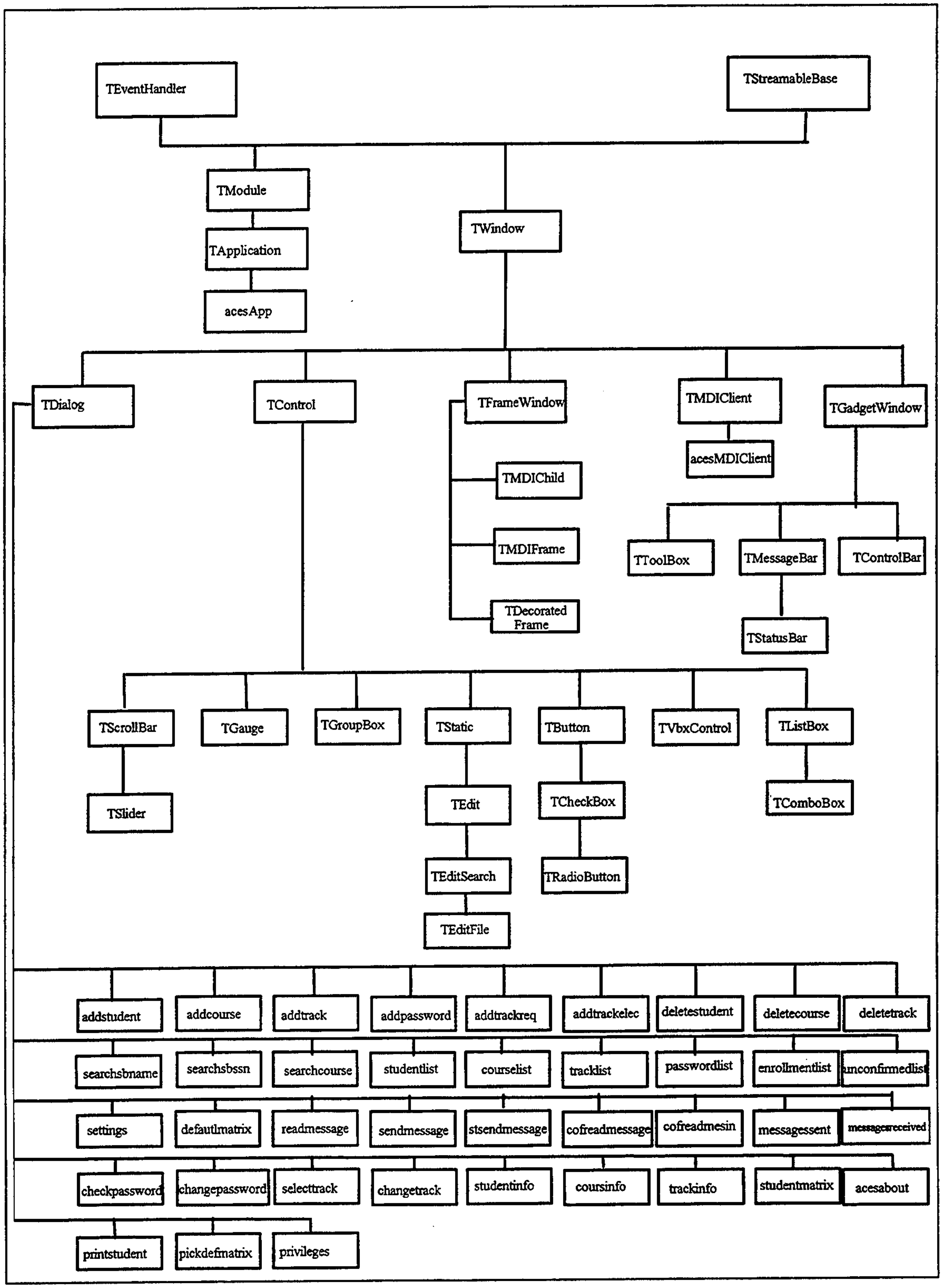


Figure 21. Hierarchical Class Diagram of ACES Application (After [Ref. 3])

- If the user logging on is a curriculum staff, then privileged access to the system is provided and the application window is displayed. If the user is a student, then only the part of the system that interests that particular student is provided by displaying an alternate application window.
- All of the paradox tables are opened for access. These tables include "Students, Courses, Tracks, Passwords, etc."
- If the user is a privileged user, then the menu and the toolbar that have the following commands are enabled: " Exit Application, Add Student, Add Course, Add Track, Add Password, Delete Student, Delete Course, Delete Track, Search Student, Search Course, Display Students, Display Courses, Display Tracks, Display Passwords, Display Settings, Display Default Matrix, Display Enrollment List, Display Students Not Confirmed, Generate Enrollment List, Send Mail, Help".
- If the user is a student, then the menu and the toolbar that have the following commands are enabled: " Exit Application, Display Student Matrix, Display Personal Info, Display Privileges, Display Mailbox, Display Courses, Display Tracks, Display Default Matrix, Search Course, Change Password, Help".

b. Multiple Document Interface (MDI) File

This is the file named "acesmdic.cpp". This file is the source file for the header file "acesmdic.h" in which the class "AcesMDIClient" is declared. This file is called from the main file "acesApp.cpp" while the application window is being constructed.

c. Database Management File

This is the file named "table.ccp". This file is the source file for the header file "table.h" in which the class "Table" is declared. Every paradox database table being opened or created is accessed through this class. Like any other C++ classes, the "Table" class also has some member variables and member functions. These variables and functions regulate the behavior of the "table object" that is created when an instance of this class is created. In the constructor of the "Table" class, the name argument provided is compared against the names of the tables already created in the directory from where the application is launched. If a table with a name matching the name argument provided to the constructor of the class is found, then this table is opened instead of creating a new table. If no table is found with the given name, then a new table is created. This file handles the following tasks:

- Creating an instance of a "Table" object.
- Opening a created or existing object.
- Accessing the paradox table being represented by the object instance.
- Searching a record in a table with a given field identification number and field data.
- Retrieving the record pointed to by the record pointer of table.
- Getting data from a field of the record currently held in the record buffer.
- Putting data to a field of the record currently held in the record buffer.
- Closing the database tables before the program terminates.

d. Dialog Box Source Files

(1) "addstdnt.cpp" is the source file for "addstdnt.h" which declares the class "addstudent". This class is derived from the OWL class "TDialog". It handles the events in "Add Student Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can enter information about a new student to be added to the "Students" table. It also displays the number of students in the database at that time. Three events are handled in this dialog box:

- Adding a new student to the database with the information displayed in the edit boxes of the dialog box at that time.
- Closing the dialog box without adding the student displayed in the edit boxes of the dialog box at that time.
- Getting online help about adding a student to the database.

(2) "addcours.cpp" is the source file for "addcours.h" which declares the class "addcourse". This class is derived from OWL class "TDialog". It handles the events in "Add Course Dialog Box". When it is called from the main file, it displays the dialog box where the curriculum staff can enter information about a new course to be added to the "Courses" table. It also displays the number of courses in the database at that time. Three events are handled in this dialog box:

- Adding a new course to the database with the information displayed in the edit boxes of the dialog box at that time.

- Closing the dialog box without adding the course displayed in the edit boxes of the dialog box at that time.
- Getting on-line help about adding a course to the database.

(3) "addtrack.cpp" is the source file for "addtrack.h" which declares the class "addtrack". This class is derived from the OWL class "TDialog". It handles the events in "Add Track Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can enter information about a new track to be added to the "Tracks" table. It also displays the number of tracks in the database at that time. Three events are handled in this dialog box:

- Adding a new track to the database with the information displayed in the edit boxes of the dialog box.
- Closing the dialog box without adding the track displayed in the edit boxes of the dialog box.
- Getting online help about adding a track to the database.

(4) "addpassw.cpp" is the source file for "addpassw.h" which declares the class "addpassword". This class is derived from the OWL class "TDialog". It handles the events in "Add Password Dialog Box". When it is called from the main file "appAces.cpp" or "passwlst.cpp" file, it displays the dialog box where the curriculum staff can enter information about a new password to be added to the "Passwords" table. It also displays the number of courses in the database at that time. Five events are handled in this dialog box:

- Adding a new password to the database with the information displayed in the edit boxes of the dialog box at that time.
- Deleting the password highlighted from the database.
- Finding information about the owner of the password highlighted.
- Closing the dialog box without adding the password displayed in the edit boxes of the dialog box at that time.
- Getting online help about adding a password to the database.

(5) "deletest.cpp" is the source file for "deletest.h" which declares the class "deletestudent". This class is derived from the OWL class "TDialog". It handles

the events in "Delete Student Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can enter the "SSN#" of the student to delete the student from the "Student" table, or, check the "Delete By Graduation Date" checkbox to delete the group of students with the specified graduation date from the "Students" table. It also displays the number of students deleted between the time the "Delete Student Dialog Box" is opened and the time it is closed. Three events are handled in this dialog box:

- Deleting a student from the database specified by "SSN#" or deleting a group of students from the database specified by "Graduation Date ".
- Closing the dialog box.
- Getting online help about deleting a student from the database.

(6) "deletcrs.cpp" is the source file for "deletcrs.h" which declares the class "deletecourse". This class is derived from the OWL class "TDialog". It handles the events in "Delete Course Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can enter the "Course No" of the course to be deleted from the "Courses" table. It also displays the number of courses deleted between the time the "Delete Course Dialog Box" is opened and the time it is closed. Three events are handled in this dialog box:

- Deleting a course specified by "Course No " entered at that time.
- Closing the dialog box.
- Getting online help about deleting a student from the database.

(7) "deletetr.cpp" is the source file for "deletetr.h" which declares the class "deletetrack". This class is derived from OWL class "TDialog". It handles the events in "Delete Track Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can either select the track from the list box and press the delete button or just double click the track name from the list box to delete it from the "Tracks" table. Three events are handled in this dialog box:

- Deleting a track highlighted in the list box from the database.
- Closing the dialog box.

- Getting online help about deleting a track from the database.

(8) "searchbn.cpp" is the source file for "searchbn.h" which declares the class "searchsbname". This class is derived from the OWL class "TDialog". It handles the events in "Search Student By Name Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can search for student information in the database by entering either the "First Name", "Last Name" or both "First Name" and "Last Name" of the student. Four events are handled in this dialog box:

- Searching for the student entered.
- Searching for the next student with the same information.
- Closing the dialog box.
- Getting online help about searching for a student by name.

(9) "searchss.cpp" is the source file for "searchss.h" which declares the class "searchsbssn". This class is derived from the OWL class "TDialog". It handles the events in "Search Student By Ssn Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can search for student information in the database by entering the "SSN#" of the student. Three events are handled in this dialog box:

- Searching the student entered.
- Closing the dialog box.
- Getting online help about searching a student by Ssn.

(10) "searchcs.cpp" is the source file for "searchcs.h" which declares the class "searchcourse". This class is derived from the OWL class "TDialog". It handles the events in "Search Course Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can search for a course from the database by entering the "Course No" of the course. Three events are handled in this dialog box:

- Searching for the course entered.
- Closing the dialog box.
- Getting online help about searching for a course.

(11) "studentl.cpp" is the source file for "student.h" which declares the class "studentlist". This class is derived from the OWL class "TDialog". It handles the events in "Student List Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can see the list of the students in the database. Six events are handled in this dialog box:

- Displaying the "Personal Information" of the selected student from the list box by opening the "Personal Info Dialog Box".
- Displaying the "Matrix" of the selected student from the list box by opening the "Student Matrix Dialog Box".
- Opening the "Send Mail Dialog Box" to send a message to the selected student.
- Printing the student list displayed.
- Closing the dialog box.
- Getting online help about "Student List"

(12) "courslst.cpp" is the source file for "courslst.h" which declares the class "courselist". This class is derived from the OWL class "TDialog". It handles the events in "Course List Dialog Box". When it is called from the main file "acesApp.cpp" or from the file "studentm.cpp", it displays the dialog box where the curriculum staff or the students can see the list of the courses in the database. Four events are handled in this dialog box:

- Displaying more information about the selected course from the list box by opening the "Course Info Dialog Box".
- Printing the course list displayed.
- Closing the dialog box.
- Getting online help about" Course List"

(13) "tracklst.cpp" is the source file for "tracklst.h" which declares the class "tracklist". This class is derived from the OWL class "TDialog". It handles the events in "Track List Dialog Box". When it is called from the main file "acesApp.cpp" or from the file "studentm.cpp", it displays the dialog box where the curriculum staff or the

students can see the list of the tracks in the database. Four events are handled in this dialog box:

- Displaying more information about the selected track from the list box by opening the "Track Info Dialog Box".
- Closing the dialog box.
- Getting online help about "Track List"

(14) "passwlst.cpp" is the source file for "passwlst.h" which declares the class "passwordlist". This class is derived from the OWL class "TDialog". It handles the events in "Password List Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can see the list of the passwords in the database. Five events are handled in this dialog box:

- Deleting the selected password from the database.
- Adding a new password to the database.
- Finding the student that has the selected password.
- Closing the dialog box.
- Getting online help about "Password List"

(15) "settings.cpp" is the source file for "settings.h" which declares the class "settings". This class is derived from the OWL class "TDialog". It handles the events in "Settings Dialog Box". When it is called from the main file "acesApp.cpp" , it displays the dialog box where the curriculum staff can see and modify the "settings" information. Four events are handled in this dialog box:

- Saving the changes made to the "Settings" information.
- Setting the "Confirmation Flags of the Students" to "False" which would tag all student records in the database to indicate that they did not confirm their next quarter's schedules.
- Closing the dialog box.
- Getting online help about "Settings"

(16) "pickdefm.cpp" is the source file for "pickdefm.h" which declares the class "pickdefmatrix". This class is derived from the OWL class "TDialog". It

handles the events in "Pick Default Matrix Type Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can see the list of the default matrices in the database. Ten events are handled in this dialog box:

- Displaying the first type of default matrix in "Default Matrix Dialog Box".
- Displaying the second type of default matrix in "Default Matrix Dialog Box".
- Displaying the third type of default matrix in "Default Matrix Dialog Box".
- Displaying the fourth type of default matrix in "Default Matrix Dialog Box".
- Displaying the fifth type of default matrix in "Default Matrix Dialog Box".
- Displaying the sixth type of default matrix in "Default Matrix Dialog Box".
- Displaying the seventh type of default matrix in "Default Matrix Dialog Box".
- Displaying the eighth type of default matrix in "Default Matrix Dialog Box".
- Closing the dialog box.
- Getting online help about "Pick Default Matrix"

(17) "enroll.cpp" is the source file for "enroll.h" which declares the class "enrollmentlist". This class is derived from the OWL class "TDialog". It handles the events in "Enrollment List Dialog Box". When it is called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can see the "enrollment list". Six events are handled in this dialog box:

- Displaying the student information of the student whose line in the list box is double clicked.
- Displaying the enrollment list for all courses enrolled.
- Displaying the enrollment list for one course whose "Course No" is entered in the edit box provided.
- Printing the enrollment list for all courses.
- Closing the dialog box.
- Getting online help about "Enrollment List".

(18) "unconlst.cpp" is the source file for "unconlst.h" which declares the class "unconfirmedlist". This class is derived from the OWL class "TDialog". It handles the events in "Students Not Confirmed Next Quarter Dialog Box". When it is

called from the main file "acesApp.cpp", it displays the dialog box where the curriculum staff can see the "list of students who did not confirm their next quarters' schedule". Four events are handled in this dialog box:

- Displaying the matrix information of the selected student from the list box.
- Printing the list displayed.
- Closing the dialog box.
- Getting online help about "Students who did not confirm their next quarters' schedules"

(19) "sendmsg.cpp" is the source file for "sendmsg.h" which declares the class "sendmessage". This class is derived from the OWL class "TDialog". It handles the events in "Send Message to Student Dialog Box". When it is called from the main file "acesApp.cpp", or from the file "studentl.cpp" it displays the dialog box, where the curriculum staff can send mail to the students and display the "Send Log" and "Receive Log". Five events are handled in this dialog box:

- Sending the message written to the specified recipient or recipients.
- Opening the "Send Log Dialog Box" to display "Send Log" information.
- Opening the "Receive Log Dialog Box" to display "Receive Log" information.
- Closing the dialog box.
- Getting online help about "Send Message to Student"

(20) "studentm.cpp" is the source file for "studentm.h" which declares the class "studentmatrix". This class is derived from the OWL class "TDialog". It handles the events in "Student Matrix Dialog Box". When it is called from the main file "acesApp.cpp", or from some other files that make calls to this file, it displays the dialog box where the curriculum staff or the student can see and modify the student matrix information. The following events are handled in this dialog box:

- Selecting a "Track" from a list of valid tracks for the student.
- Changing the "Student's Track" selected before.
- Changing the "Password" for logging on to the system.
- Displaying the list of "Courses" in the database.

- Displaying the list of "Tracks" in the database.
- Displaying the "Default Matrix".
- Displaying the "Privileges".
- Displaying the "Mailbox".
- Saving the changes made to the matrix.
- Displaying the matrix information that was in the database when the "Student Matrix Dialog Box" was opened, disregarding the changes made to the matrix.
- Confirming next quarters schedule.
- Denying next quarter's schedule.
- Closing the dialog box.
- Getting online help about "Student Matrix".

(21) "studenti.cpp" is the source file for "studenti.h" which declares the class "studentinfo". This class is derived from the OWL class "TDialog". It handles the events in "Student Info Dialog Box". When it is called from the main file "acesApp.cpp", or from some other files that make calls to this file, it displays the dialog box where the curriculum staff or the students can see information about the student. The curriculum staff can also modify the displayed information. The following events are handled in this dialog box:

- Going to the first student record in the students table. (For Curriculum Staff)
- Going five records back in the students table. (For Curriculum Staff)
- Going to the previous student record in the students table. (For Curriculum Staff)
- Going to the next student record in the students table. (For Curriculum Staff)
- Going five records forward in the students table. (For Curriculum Staff)
- Going to the last student record in the database. (For Curriculum Staff)
- Saving the changes made to the student information displayed. (For Curriculum Staff)
- Deleting the displayed student from the database. (For Curriculum Staff)
- Displaying the matrix of the student by opening "Student Matrix Dialog Box".

- Closing the dialog box.
- Getting online help about "Send Message to Student"

(22) "privileg.cpp" is the source file for "privileg.h" which declares the class "privileges". This class is derived from the OWL class "TDialog". It handles the events in "Privileges Dialog Box". When it is called from the main file "acesApp.cpp", or from the file "studentm.cpp" it displays the dialog box where the curriculum staff or students can see the "Privileges" information. The curriculum staff can also modify the information displayed. Three events are handled in this dialog box:

- Saving the changes made to the information displayed.
- Closing the dialog box.
- Getting online help about "Privileges".

(23) "readmsg.cpp" is the source file for "readmsg.h" which declares the class "readmessage". This class is derived from the OWL class "TDialog". It handles the events in "Messages from Curricular Officer Dialog Box". When it is called from the main file "acesApp.cpp", or from the file "studentm.cpp" it displays the dialog box where the students can see their "mailbox" information. The curriculum staff can also see this information. Five events are handled in this dialog box:

- Deleting the first message displayed.
- Deleting the second message displayed.
- Sending mail to the curricular officer.
- Closing the dialog box.
- Getting online help about "Messages from Curricular Officer".

(24) "defmatri.cpp" is the source file for "defmatri.h" which declares the class "defaultmatrix". This class is derived from the OWL class "TDialog". It handles the events in "Default Matrix Dialog Box". When it is called from the main file "acesApp.cpp", or from the file "pickdefm.cpp" it displays the dialog box where the curriculum staff or students can see the "Default Matrix" information. The curriculum staff can also modify the information displayed. Five events are handled in this dialog box:

- Saving the changes made to the information displayed.

- Displaying the default matrix information that was in the database when the "Default Matrix Dialog Box" was opened, disregarding the changes made to the matrix.
- Printing the default matrix displayed.
- Closing the dialog box.
- Getting online help about "Default Matrix".

(25) "chanpass.cpp" is the source file for "chanpass.h" which declares the class "changepassword". This class is derived from the OWL class "TDialog". It handles the events in "Change Password Dialog Box". When it is called from the main file "acesApp.cpp", or from the file "studentm.cpp" it displays the dialog box where the students can change their existing passwords. The curriculum staff can also change a student's password through this dialog box. Three events are handled in this dialog box:

- Changing the password to the one entered in the dialog box.
- Closing the dialog box.
- Getting online help about "Change Password".

(26) "checkpas.cpp" is the source file for "checkpas.h" which declares the class "checkpassword". This class is derived from the OWL class "TDialog". It handles the events in "Default Matrix Dialog Box". When the ACES application is started, this dialog box is the first thing that is displayed in the application. The user is prompted to enter the SSN# (Social Security Number) and his password. This way, the access of the unauthorized users' access to the system is prevented.

(27) "stsendms.cpp" is the source file for "stsendms.h" which declares the class "stsendmessage". This class is derived from the OWL class "TDialog". It handles the events in "Default Matrix Dialog Box". When it is called from the file "readmsg.cpp", it displays the dialog box where the students can send message to the curricular officer. Two events are handled in this dialog box:

- Sending the message written in the dialog box.
- Closing the dialog box.

(28) "printstd.cpp" is the source file for "printstd.h" which declares the class "printstudent". This class is derived from the OWL class "TDialog". It handles the events in "Print Student List Dialog Box". When it is called from the file "studentl.cpp", it displays the dialog box where the curriculum staff can see the list of options for the layout of the print out of the "Student List". Three events are handled in this dialog box:

- Printing the "Student List" with the option selected.
- Closing the dialog box without printing anything.
- Getting online help about "Print Student List".

(29) "mesrecv.cpp" is the source file for "mesrecv.h" which declares the class "messagesreceived". This class is derived from the OWL class "TDialog". It handles the events in "Messages Received Dialog Box". When it is called from the file "sendmsg.cpp", it displays the dialog box where the curriculum staff can see the list of messages received from the students. Six events are handled in this dialog box:

- Opening the "Reply Dialog Box" to write a reply message to the selected received message.
- Displaying the "Student Information" for the selected received message.
- Displaying the contents of the selected received message.
- Deleting the selected received message from the "Receive Log".
- Closing the dialog box.
- Getting online help about "Messages Received from the Students".

(30) "messent.cpp" is the source file for "messent.h" which declares the class "messagessent". This class is derived from the OWL class "TDialog". It handles the events in "Messages Sent Dialog Box". When it is called from the file "sendmsg.cpp", it displays the dialog box where the curriculum staff can see the list of the messages sent to the students. Five events are handled in this dialog box:

- Displaying the "Student Information" for the selected sent message.
- Displaying the contents of the selected sent message.
- Deleting the selected sent message from the "Send Log".

- Closing the dialog box.
- Getting online help about "Messages Sent to the Students".

(31) "cofredms.cpp" is the source file for "cofredms.h" which declares the class "cofreadmes". This class is derived from the OWL class "TDialog". It handles the events in "Message Sent to Student Dialog Box". When it is called from the file "messent.cpp", it displays the dialog box where the curriculum staff can see the contents of a message sent to a student. Closing of the dialog box is the only event handled in this dialog box.

(32) "cofreadi.cpp" is the source file for "cofreadi.h" which declares the class "cofreadmesin". This class is derived from the OWL class "TDialog". It handles the events in "Message Received from Student Dialog Box". When it is called from the file "mesrecv.cpp", it displays the dialog box where the curriculum staff can see the contents of a message received from a student. Closing of the dialog box is the only event handled in this dialog box.

(33) "addtrace.cpp" is the source file for "addtrace.h" which declares the class "addtrackelec". This class is derived from the OWL class "TDialog". It handles the events in "Add Track Elective Dialog Box". When it is called from the file "addtrack.cpp" or from "trackinfo.cpp", it displays the dialog box where the curriculum staff can enter information about a new "Track Elective Course" to the "Track" from whose "Track Information" dialog box the "Add Track Elective" dialog box is opened. Two events are handled in this dialog box:

- Adding the entered "Track Elective Course" to the list of "Track Electives" for the selected track.
- Closing the dialog box.

(34) "addtrreq.cpp" is the source file for "addtrreq.h" which declares the class "addtrackreq". This class is derived from the OWL class "TDialog". It handles the events in "Add Track Requirement Dialog Box". When it is called from the file "addtrack.cpp" or from "trackinfo.cpp", it displays the dialog box where the curriculum staff can enter information about a new "Track Requirement Course" to the "Track" from

whose "Track Information" dialog box the "Add Track Requirement" dialog box is opened. Two events are handled in this dialog box:

- Adding the entered "Track Requirement Course" to the list of "Track Requirements" for the selected track.
- Closing the dialog box.

(35) "acesabot.cpp" is the source file for "acesabot.h" which declares the class "acesabout". This class is derived from the OWL class "TDialog". When it is called from the main file "acesApp.cpp", it displays the dialog box, where the users can get version and programmer information about "ACES".

e. Paradox Database Files

These are the files with ".db" and ".px" extensions. Files with ".db" extensions are the "Paradox Database Files" where the information about the entities in "ACES" application is stored and files with ".px" extensions are the index files for the database files. So there is one ".px" file for each ".db" file. These files are used during "run time", so they are not needed at compilation time. If at "run time" these files are not present, then they are created by the application with no records in them. The following is the list of those files:

- "courses.db" and "courses.px" are for courses.
- "defaultm.db" and "defaultm.db" are for default matrices.
- "enroll.db" and "enroll.px" are for the enrollment list.
- "mesrecv.db" and "mesrecv.px" are for the messages received from the students.
- "messent.db" and "messent.px" are for the messages sent to the students.
- "password.db" and "password.px" are for the passwords.
- "settings.db" and "settings.px" are for the settings.
- "students.db" and "students.px" are for the students.
- "t1cores.db" and "t1cores.px" are for the core courses of "track no 1".
- "t2cores.db" and "t2cores.px" are for the core courses of "track no 2".
- "t3cores.db" and "t3cores.px" are for the core courses of "track no 3".

- "t4cores.db" and "t4cores.px" are for the core courses of "track no 4".
- "t5cores.db" and "t5cores.px" are for the core courses of "track no 5".
- "t6cores.db" and "t6cores.px" are for the core courses of "track no 6".
- "t7cores.db" and "t7cores.px" are for the core courses of "track no 7".
- "t8cores.db" and "t8cores.px" are for the core courses of "track no 8".
- "telects.db" and "telects.px" are for the "Elective Courses" for all tracks.
- "tracks.db" and "tracks.px" are for the tracks.

f. Print Files

These are the files with ".rpt" extension. These files provide the format for the lists to be printed. They are used at "run time", so they are not needed at compilation time. But they have to be present at "run time", otherwise the application will not print anything when it is asked to do so. The following is the list of those files:

- "crslist.rpt" is for course list.
- "defmat1.rpt" is for the first type of default matrix.
- "defmat2.rpt" is for the second type of default matrix.
- "defmat3.rpt" is for the third type of default matrix.
- "defmat4.rpt" is for the fourth type of default matrix.
- "defmat5.rpt" is for the fifth type of default matrix.
- "defmat6.rpt" is for the sixth type of default matrix.
- "defmat7.rpt" is for the seventh type of default matrix.
- "defmat8.rpt" is for the eighth type of default matrix.
- "enroll.rpt" is for the enrollment list.
- "notconf.rpt" is for the list of students who did not confirm.
- "stdlist1.rpt" is for the first type of students list.
- "stdlist2.rpt" is for the second type of students list.
- "stdlist3.rpt" is for the third type of students list.
- "stdlist4.rpt" is for the fourth type of students list.

g. Help Utility Related Files

These are the files with ".hpj", ".hlp", and ".rtf" extensions. The files with ".rtf" extension are called "Rich Text Format" files. These files are created by any wordprocessor that supports "Rich Text Format". The file with ".hpj" extension is the file that gives information to the "Windows Help Compiler" about the ".rtf" files making up the "Help File" with ".hlp" extension. The ".hpj" and ".hlp" files bear the same name. The "Windows Help Compiler 'hc31.exe' " takes the ".hpj" and ".rtf" files as input and creates the "Windows Help File" with ".hlp" extension. The files used in the ACES application to create the file "aces.hlp" are "aces.hlp", "dlgboxes.rtf", "keyboard.rtf", "keys.rtf", "mainhelp.rtf" and "toolbar.rtf".

h. Object Files

These are the files with ".obj" extension. These files are created as a result of "C++ compiling" of the ".cpp" files. There is one ".obj" file for each ".cpp" file after a full compilation.

i. Bitmap Files

These are the files with ".bmp" extension. These files are used in "aces.hlp" file to put the pictures of toolbar icons to the help file created.

j. Other Files

(1) "test3.apx" is the database file for the "AppExpert" source. This file is required when editing files in "Class Expert" in programming environment.

(2) " acesapp.def" is the "ascii" text file including some technical information about the application. This file needs not to be edited manually, since the programming environment, itself, is taking care of this file.

(3) "aces.dsw" is an application related binary file and it is created by the compilation and linking process. It needs not to be edited by the programmer.

(4) "aces.ide" is the project file for the ACES application. It is created when the ACES application project file was first created. It holds information

regarding the files in the project. When the programmer wants to edit the source files, he has to open this file in Borland C++ Programming Environment.

(5) "aces.map" is the "map file" created by the linking process and it is useful to the programmer in debugging the code. It is optional to create a map file.

(6) "acesapp.rc" is the resource script file. In this file, there are the scripts for the dialog boxes, icons, bitmaps, string tables, menus, menu accelerators etc.

(7) "acesapp.rh" is the resource header file. In this file, there are the identifiers regarding the objects used in dialog boxes or in other windows tools.

(8) "acesapp.res" is a binary file regarding the ACES application. It is created when the ".rc" file is compiled and used when the linker is creating the "executable" file with ".exe" extension.

(9) "acesapp.rws" is another application related file and automatically updated by the compilation and linking process.

(10) "aces.ph" and "aces.obr" are application related files and automatically updated by the compilation and linking process.

(11) "aces.exe" is the executable file created by the linking process. Figure 22 shows how an executable file is created in "OWL (Object Windows Library) Windows Application" development.

3. Interaction Of Classes At Run Time

The ACES application is started by the main file named "acesApp.cpp". This file makes calls to some of the source files during the execution of the application. This file can be perceived as being at the control level of the execution graph. During regular session when a user runs the application, the program execution graph starts at this control level and goes to different branches depending on what the user is doing in the program. Opening dialog boxes one after another is analogous to going to different nodes in the execution graph. The user has to come back to the control level to be able to quit the application. There is no other way of quitting the application. There are two different execution graphs depending on the person logged on. The execution graph for curriculum staff is shown in Figure 23, and the one for students is shown in Figure 24.

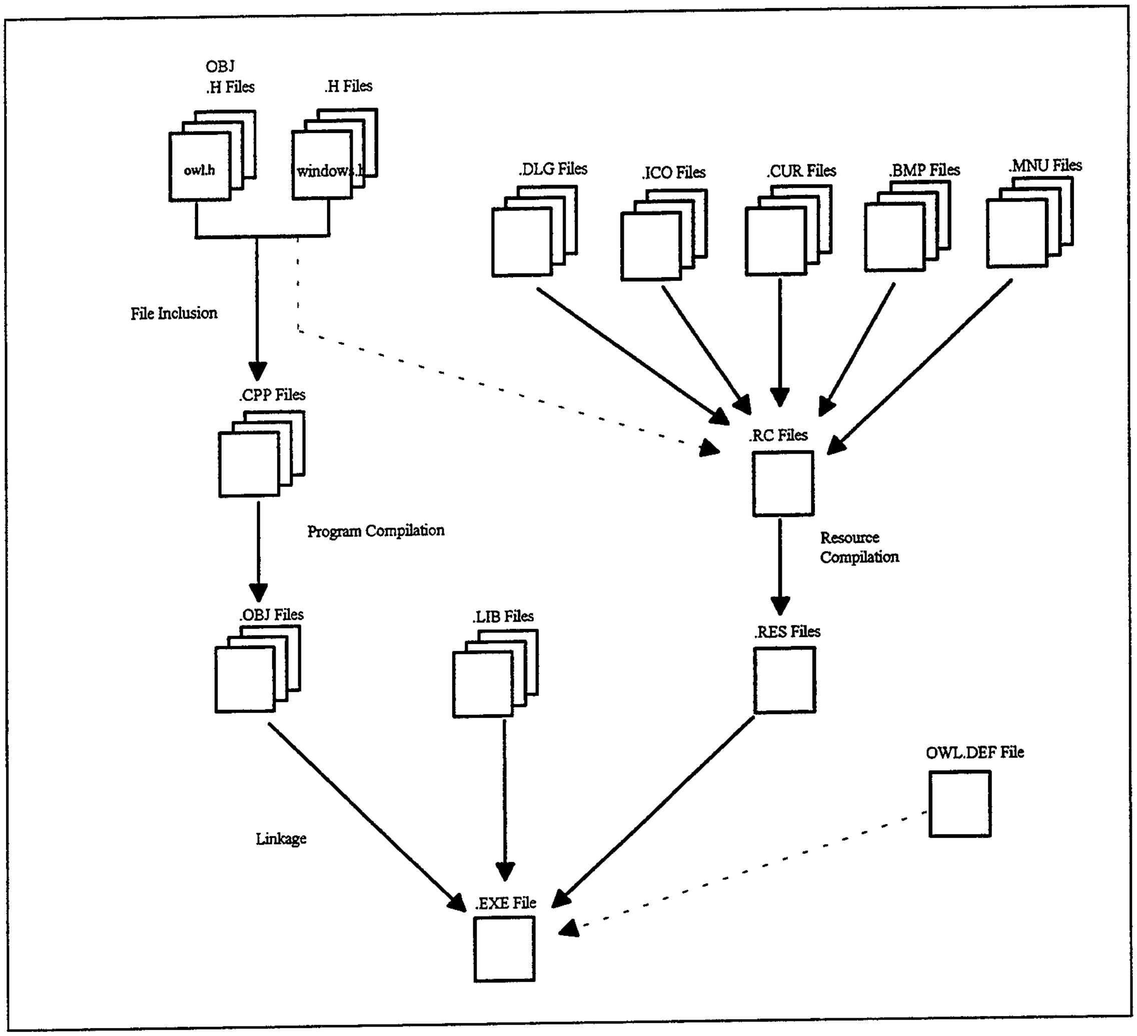


Figure 22. From [Ref. 4] The Process of Getting an Executable File in OWL Windows Development

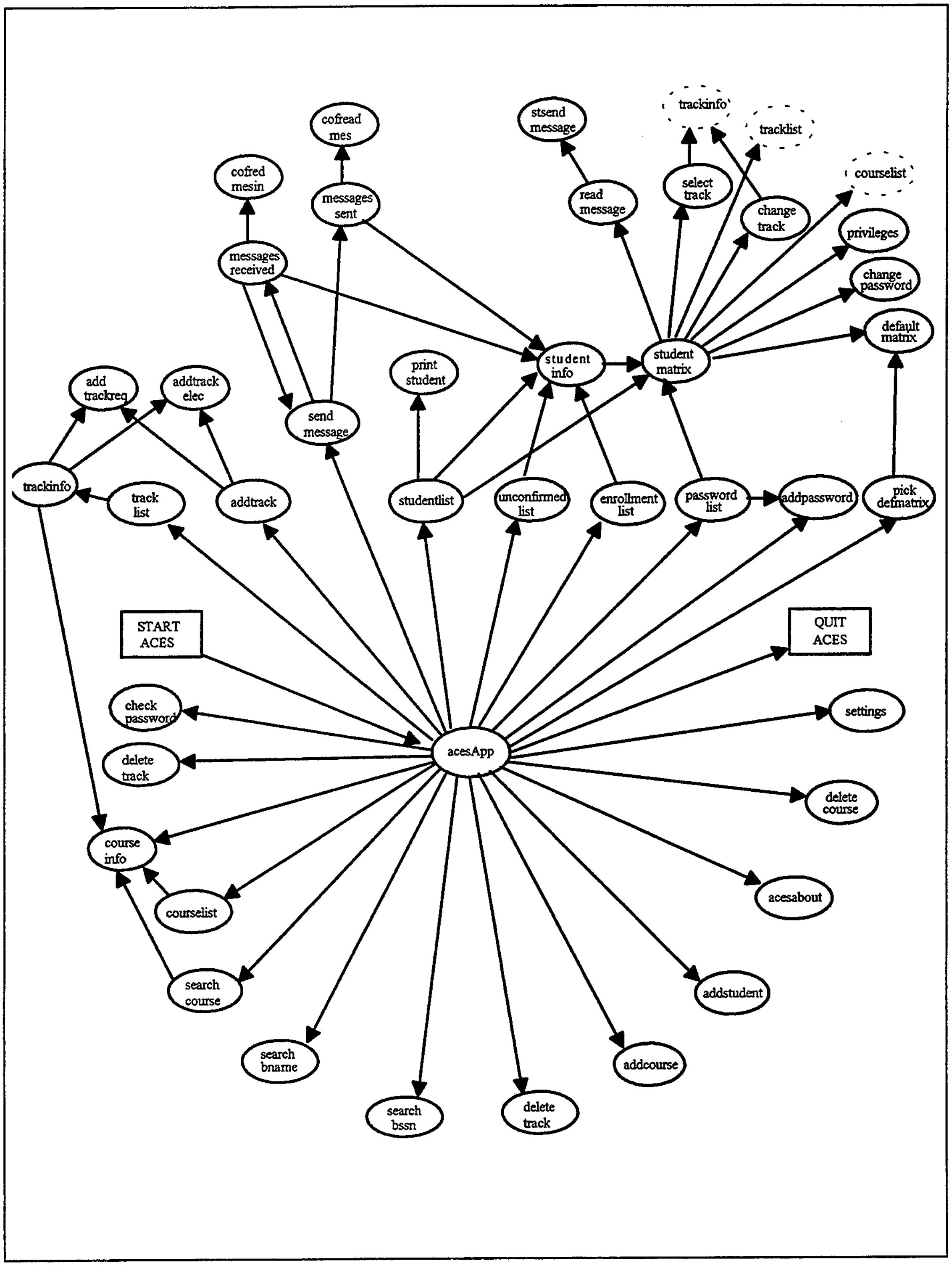


Figure 23. Execution Graph for Curriculum Staff

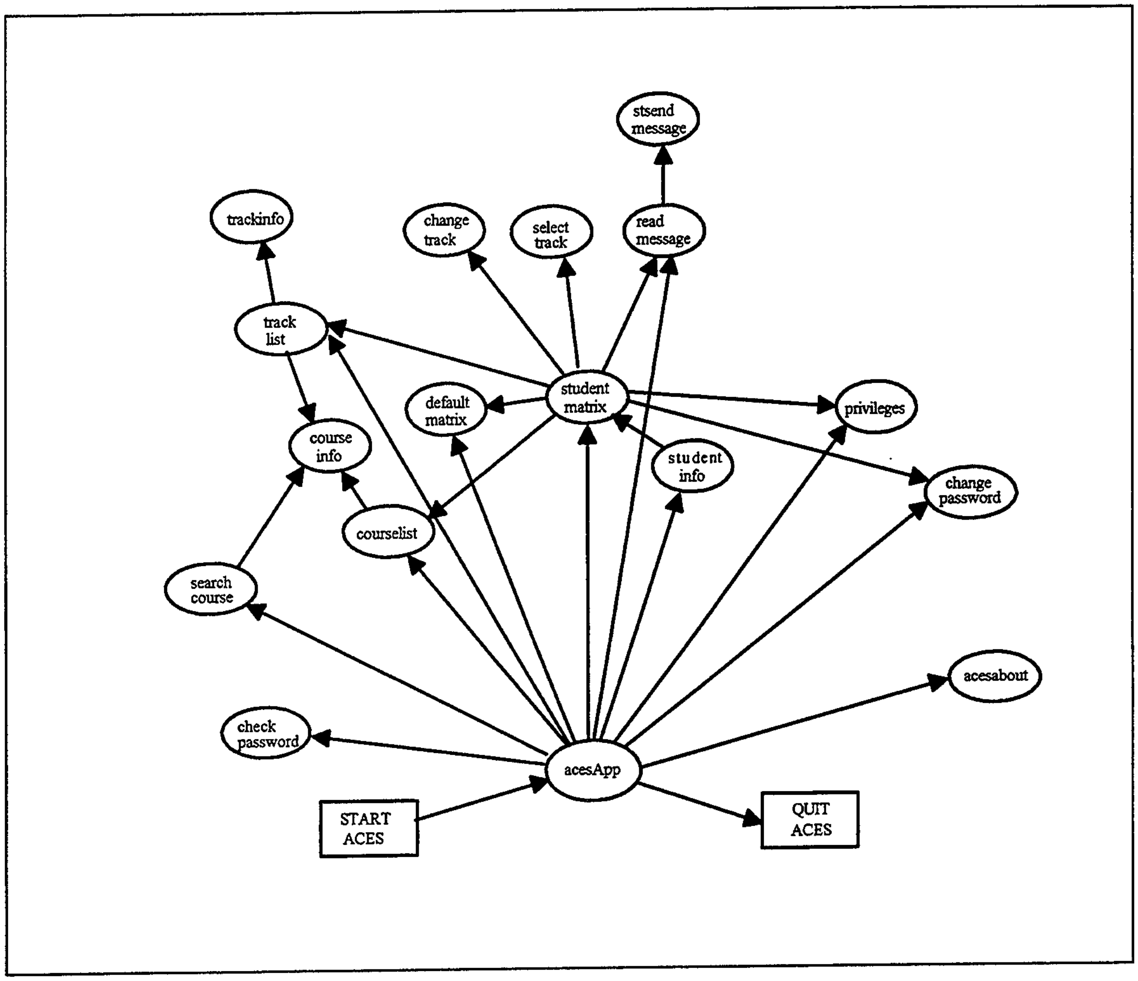


Figure 24 . Execution Graph for Students

E. SYSTEM REQUIREMENTS

Since the ACES is a multi-user application, an executable file named "share.exe" that comes usually with the "MS-DOS" operating system and is usually put under the directory "C:\dos", must be executed before the execution of the "aces.exe" executable file. The best way is to execute this file by putting the line "C:\dos\share.exe" in "autoexec.bat" file before any statement that starts the "Windows Environment".

The ACES is designed to work on "Windows 3.1" or higher. On "Windows 3.0" and lower, it will not work. If other operating systems like "OS/2" supports windows applications, the ACES will work on these operating systems too.

Other than this, following files have to be in the directory "C:\aces" to make the program run.

1. Executable File

This is the file named "aces.exe". The user can create a "program group" in program manager and put this file as a group item in this program group. This way the "ACES" application can be started right from the "Program Manager".

2. Dynamic Library Files

These are the files with ".dll" extension. These are not the files just created for this application, but the files that come with the applications and a programmer may use in the application that he is developing. These files are required to run an application developed by using the software tools with which they came. For that reason, there are some "DLL" files that the ACES application needs to have in the directory where the executable file is located. These are all "runtime" libraries which means that these files are needed at run time. Following is the list of these "DLL" files that the ACES application needs:

- "bc40rtl.dll".
- "bids40.dll".
- "owl200.dll".
- "pxengwin.dll".
- "crpe.dll".
- "pdbpdx.dll".

3. Paradox Database Files

These are the files with ".db" and ".px" extensions. As explained in the previous section, ".db" files are the actual "Paradox Database Files" that store the information about entities like "Students", "Courses", "Default Matrix", ... The files with ".px" extension are the "index" files that keep the information about the type and the length of the fields of a record in the corresponding ".db" files. Following is the list of database files:

- "courses.db" and "courses.px".
- "defaultm.db" and "defaultm.px".
- "enroll.db" and "enroll.px".
- "mesrecv.db" and "messent.px".
- "password.db" and "password.px".
- "settings.db" and "settings.px".
- "students.db" and "students.px".
- "t1cores.db" and "t1cores.px".
- "t2cores.db" and "t2cores.px".
- "t3cores.db" and "t3cores.px".
- "t4cores.db" and "t4cores.px".
- "t5cores.db" and "t5cores.px".
- "t6cores.db" and "t6cores.px".
- "t7cores.db" and "t7cores.px".
- "t8cores.db" and "t8cores.px".
- "telects.db" and "telects.px".
- "tracks.db" and "tracks.px".

4. Crystal Reports Print Files

These are the files with ".rpt" extension and required at "runtime". Following is the list of these files:

- "crslist.rpt".
- "defmat1.rpt".
- "defmat2.rpt".
- "defmat3.rpt".
- "defmat4.rpt".
- "defmat5.rpt".

- "defmat6.rpt".
- "defmat7.rpt".
- "defmat8.rpt".
- "enroll.rpt".
- "notconf.rpt".
- "stdlist1.rpt".
- "stdlist2.rpt".
- "stdlist3.rpt".
- "stdlist4.rpt".

5. Help File

This is the file named "aces.hlp" and required at run time.

F. PROBLEMS ENCOUNTERED

The biggest problem was the need for a bigger "RAM" for the computer in which this application was compiled and linked. The size of "aces.exe" is approximately 2.8 MB. Most probably for this reason, when the application was being compiled and linked originally with an "8 MB RAM", there were errors stopping compiling and linking, although there were no errors when the executible size was not more than 1 MB during the development of the application.

The solution was to increase the RAM size, and so was it done. The RAM Memory Size was upgraded to 16 MB and problems defined above ceased to exist.

The other problems encountered were usually programming language and programming tools related. Therefore, the obvious solution was to learn using the language and the tools more proficiently.

IV. SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

A. SUMMARY

The ACES application is a new approach to provide better means to both the curriculum staff and the students in the Computer Science Curriculum to deal with the process of "Course Enrollment". Its user-friendly environment, not only makes the process faster but also reduces the time spent due to errors that happen as a result of many manual activities involved in the current system. It enhances the band of communication between the curriculum staff and the students. To summarize, the ACES application accomplishes the following main tasks:

- It keeps information about students, courses, tracks and making it available to the authorized users.
- It enables the curriculum staff to control the information kept in the system.
- It enables the curriculum staff to generate enrollment list to be sent to the departments.
- It enables the students to edit their own matrices and enroll for courses.
- It enables both the curriculum staff and students to send messages to each other.

B. CONCLUSIONS

Automating a process in any kind of system can be done in many different ways. Most of the time, there is no "best". In other words, there are always things that can be improved. The most important criterion is the user satisfaction. In this regard, ACES application has gained the approval of the advisor of this research and the curriculum staff. Although it could not find the opportunity to present itself to a large community of users, it also got the positive remarks of some group of students in the Computer Science Curriculum.

The design of the automation of a system is not only done at the beginning. Instead it is an ongoing process. Therefore, it never stops until the automation is given its final shape. It was the same way what was practiced in the ACES application development. The design of the ACES application was updated depending on the feedback obtained from the regularly hold meetings with the curriculum staff and the thesis advisor. So, as a result, it came to a point where it was a lot more sophisticated , functional and error free than it was originally designed or thought to be.

C. RECOMMENDATIONS FOR FUTURE RESEARCH

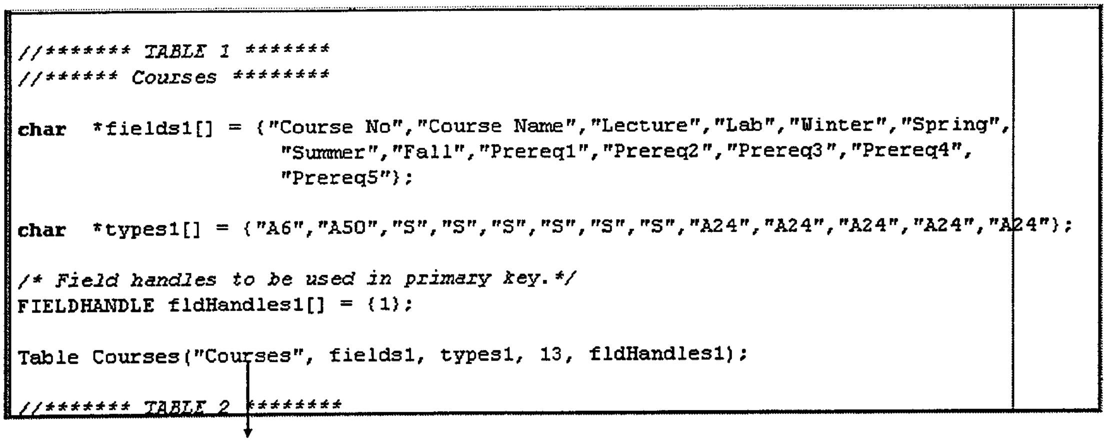
Because of the time considerations, the ACES application was not tested in the in a network environment. A possible future study could be testing it in the real time and deriving results from it to be used for further development or correction. If the ACES applications is to be used in a network environment of PCs, the following suggestions are worth looking at:

The ACES application is a Windows application designed to work in Microsoft Windows Version 3.1 or later operating environments. To make it run in a network environment like UNIX, the application has to be rewritten using an application development tool that allows the creation of client-server network applications.

The "Microsoft Windows For Workgroups Version 3.11" is a very good choice for making the ACES application run in a network environment without rewriting the current code. Windows for Workgroups can connect the personal computers in a network environment. These computers can share files and directories within this network. When the ACES application starts, it looks for the database files in "C:\aces" directory. This creates a problem if multiple copies of the application is started in different PCs. Because, each PC will have their own database files which will, as a result, create a data inconsistency in the system. A similar problem will occur when a user wants to print some data from the application. The solution to this problem is to give each PC in the network their own copies of the ACES application but keep only one copy of the database files. One of the PCs in the network, most probably the PC used by the Curricular Officer, can have these shared database files in its "C:\aces" directory and the other PCs can access this directory to read from or write to the database files. In this case, the "C:\aces" directory has to be declared "shared" and the other PCs in the network have to define this shared directory in their "File Manager" application by assigning it a drive letter. Now, the question "How will the application know which directory to look at to find the database files?" comes to mind. The solution that we are going to explain is to hard code the path of the database files in the calls to the Paradox Database Engine function "PXTableCreate". In "acesApp.cpp" file, the creation of the instances of the database files are realized and the names of the database files to be opened are specified here. In a stand-alone system, just the name of the database file will be sufficient, because the database files are going to be in the same directory as the executable file. For the other copies of the ACES application running on other PCs, the path has to specify the shared drive. For that reason, to serve as an example for future study, we designated the shared drive to be "J:\" in all PCs in the network and recompiled the code to get a slightly different version of the ACES application. In this version of the source code, a "J:\\" expression is added to the beginning of all database file names in initialization expressions in "acesApp.cpp" file. It is important to note here that a double backslash (\\) is required as the directory divider in the path expression. This is a C language syntax rule.

Another thing that has to be done is to rewrite the print files that have a ".rpt" extension in "C:\aces" directory using the "Crystal Reports", the same tool used for creating all of the print files in the ACES application. The reason for rewriting these files is that, when the user of the ACES application presses a "Print" button, the application looks for the database files which will be in the server computer. The user of the "Crystal Reports" is prompted to enter the location of the database files before he can create a print file. Therefore the print files in the client computers have to be modified to change the location of the database files from "C:\aces" to "J:\\aces". The "J" drive is the example shared network drive in our study.

One of the database initialization calls in "acesApp.cpp" file for the first type of the ACES application is shown in Figure 25. This application can be seen as the server application in the network and the unique copies of the database files will be kept in the machine in which this application will be executed.

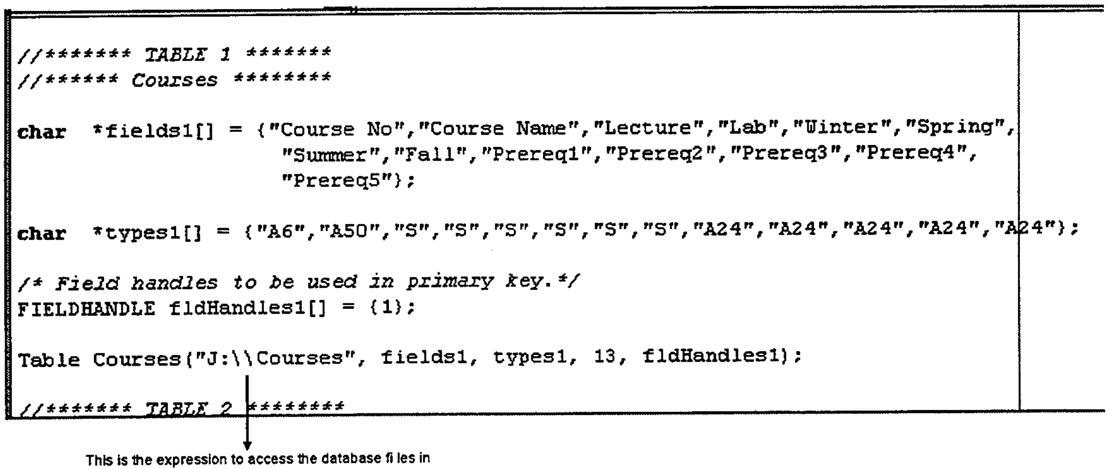


```
//******* TABLE 1 *******
//****** Courses ********

char  *fields1[] = {"Course No","Course Name","Lecture","Lab","Winter","Spring",
                    "Summer","Fall","Prereq1","Prereq2","Prereq3","Prereq4",
                    "Prereq5"};

char  *types1[] = {"A6","A50","S","S","S","S","S","S","A24","A24","A24","A24","A24"};

/* Field handles to be used in primary key.*/
FIELDHANDLE fldHandles1[] = {1};

Table Courses("Courses", fields1, types1, 13, fldHandles1);

//******* TABLE 2 ********
```

This is the expression to access the database files in the PC, where the unique copies of all of the database files are kept

Figure 25. Sample Code for the First Type of the "aces.exe" Executable File

One of the database initialization calls in "acesApp.cpp" file for the second type of the ACES application is shown in Figure 26 . This application can be seen as the client application in the network and no database files will be kept in the machine in which this application will be executed.



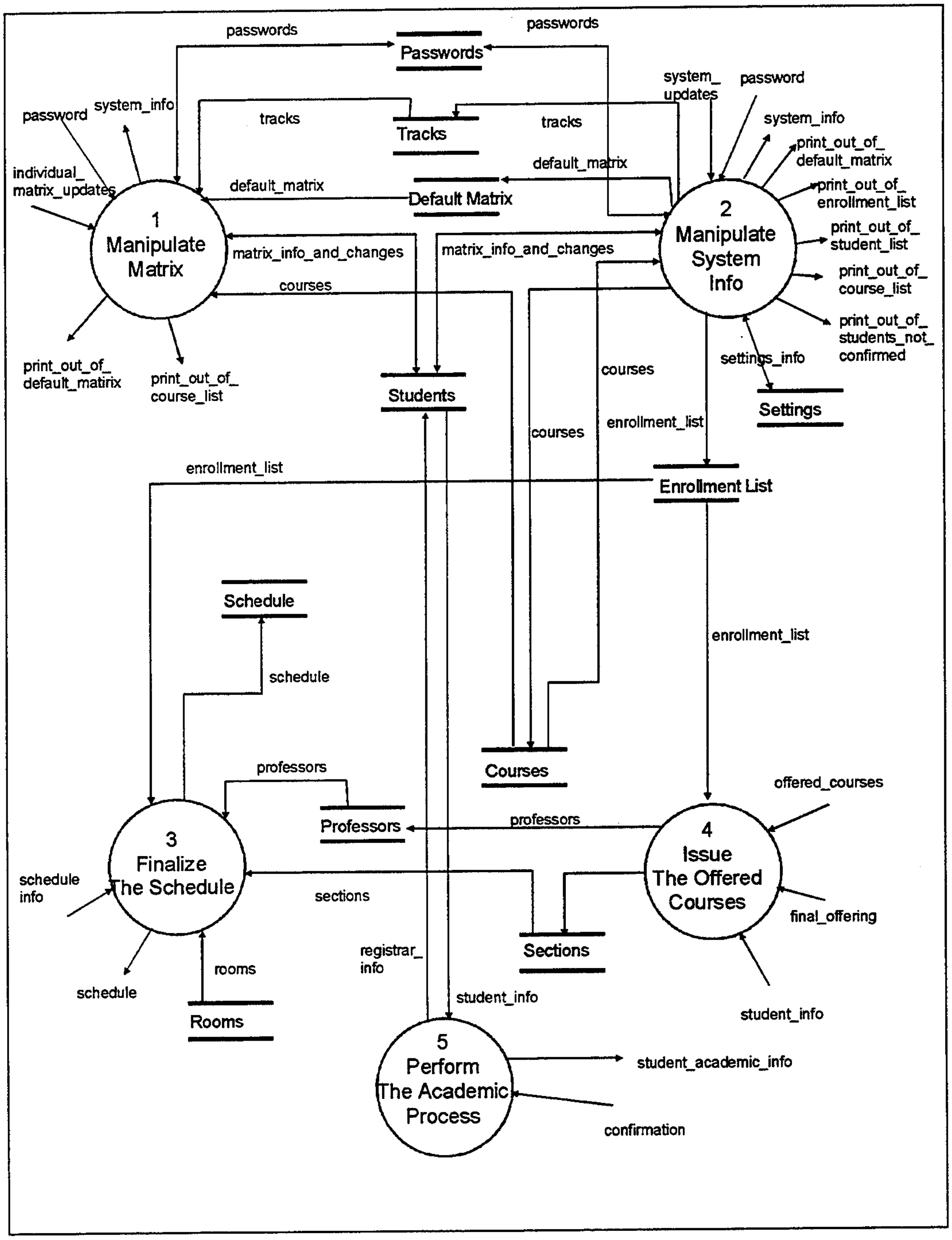
```
//******* TABLE 1 *******
//****** Courses ********

char  *fields1[] = {"Course No","Course Name","Lecture","Lab","Winter","Spring",
                    "Summer","Fall","Prereq1","Prereq2","Prereq3","Prereq4",
                    "Prereq5"};

char  *types1[] = {"A6","A50","S","S","S","S","S","S","A24","A24","A24","A24","A24"};

/* Field handles to be used in primary key.*/
FIELDHANDLE fldHandles1[] = {1};

Table Courses("J:\\Courses", fields1, types1, 13, fldHandles1);

//******* TABLE 2 ********
```

This is the expression to access the database fi les in the shared directory which is specified as "J:\" here.

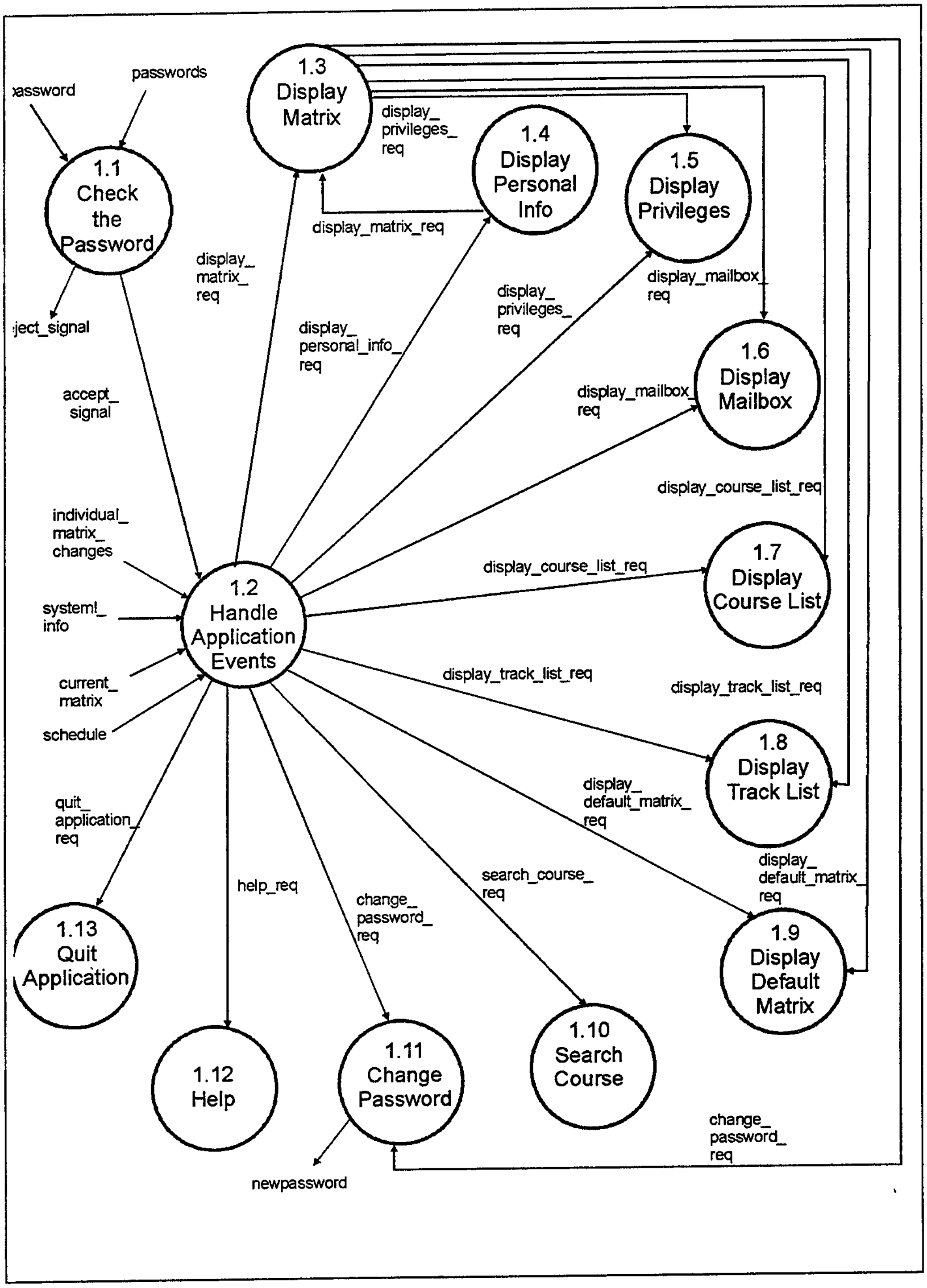
Figure 26. Sample Code for the Second Type of the "aces.exe" Executable File

Further work involving the study about ACES application could be to enhance the capabilities of the system by making it more functional. As an example, the information about students and courses coming from other offices like "Registrar" in the school can be automatically fed into the system instead of entering these information manually as in the current case.

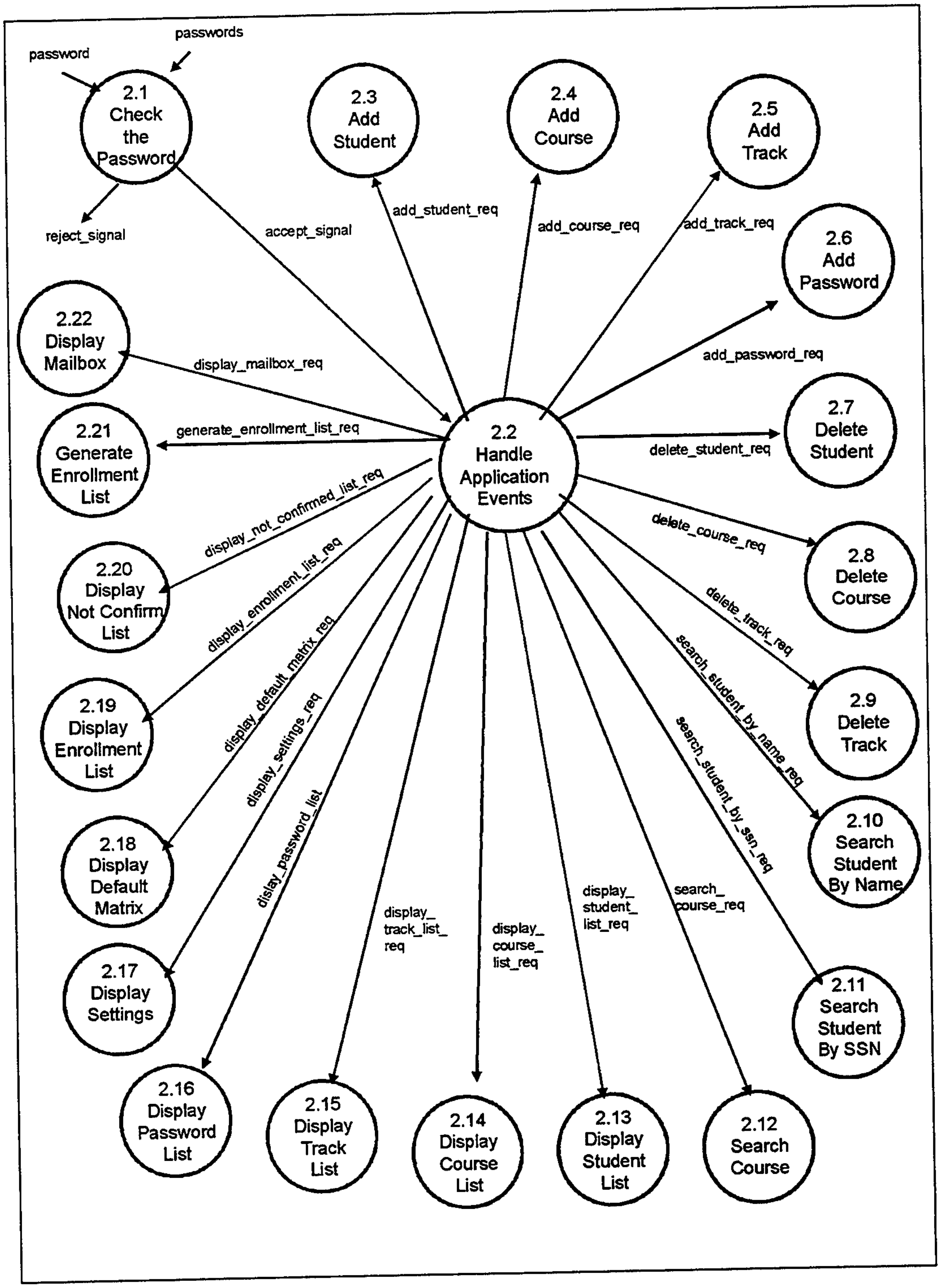
APPENDIX A.DATA FLOW DIAGRAMS



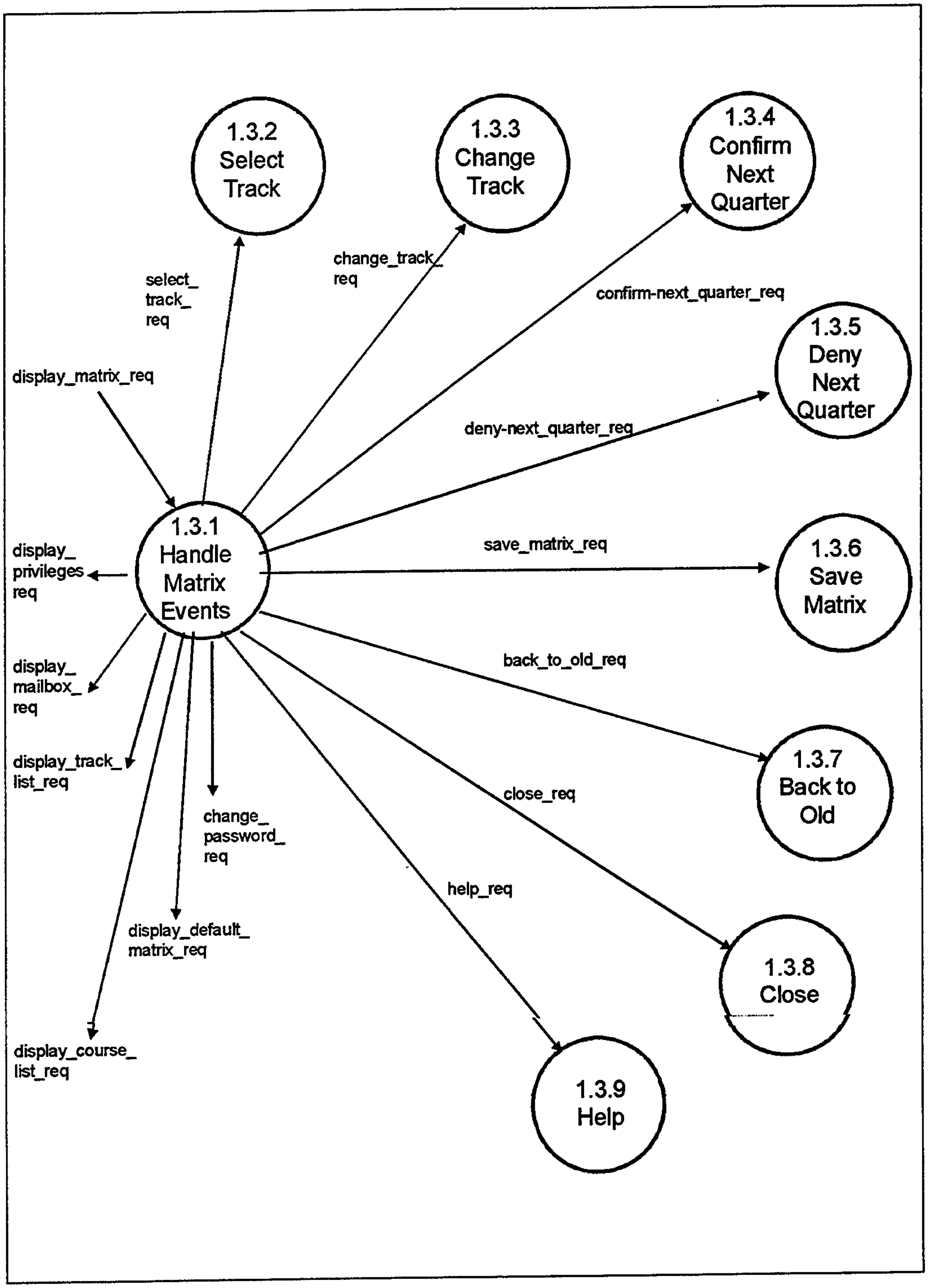
LEVEL 0



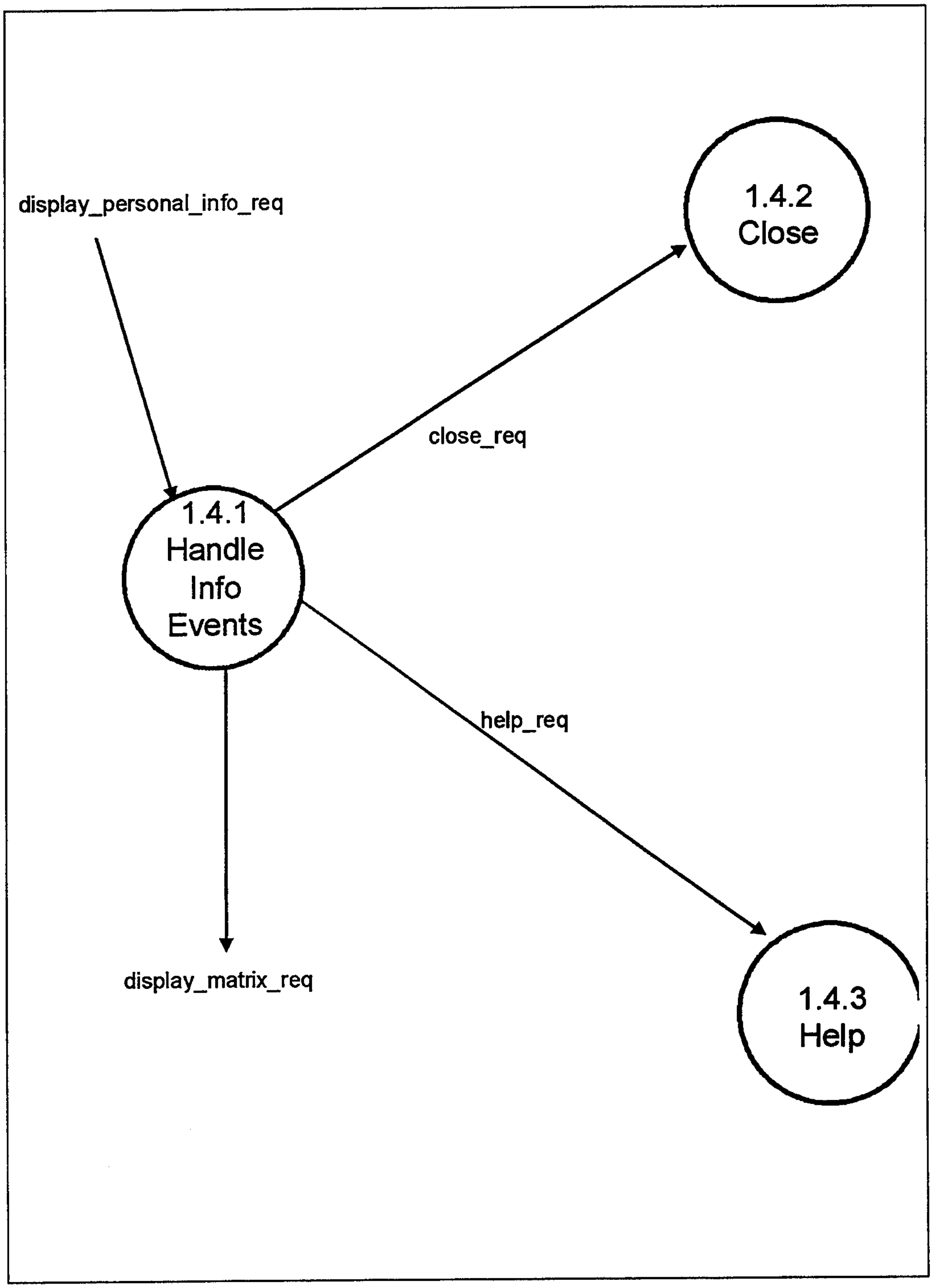
LEVEL1 (Process 1)



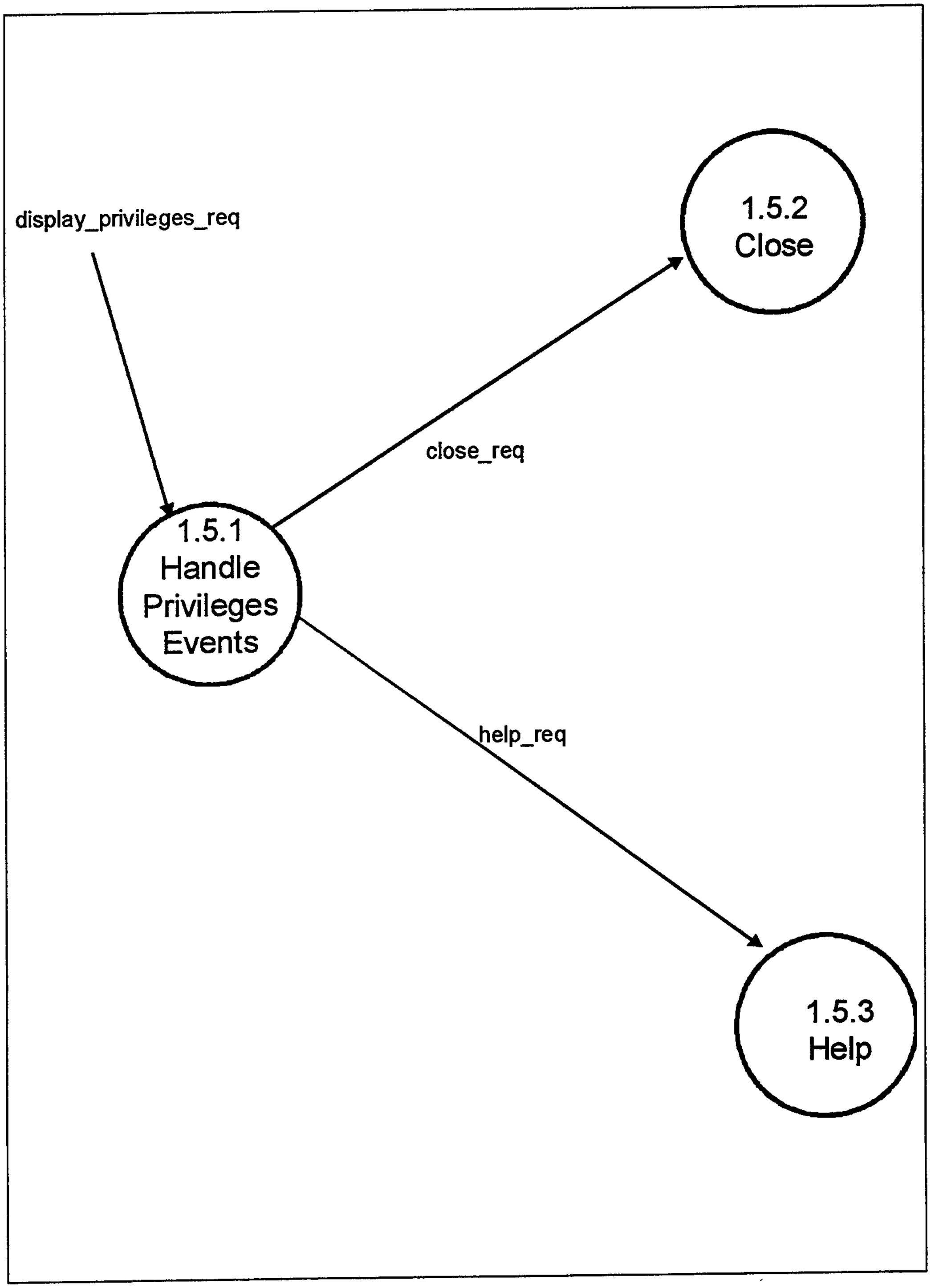
LEVEL 1 (Process 2)



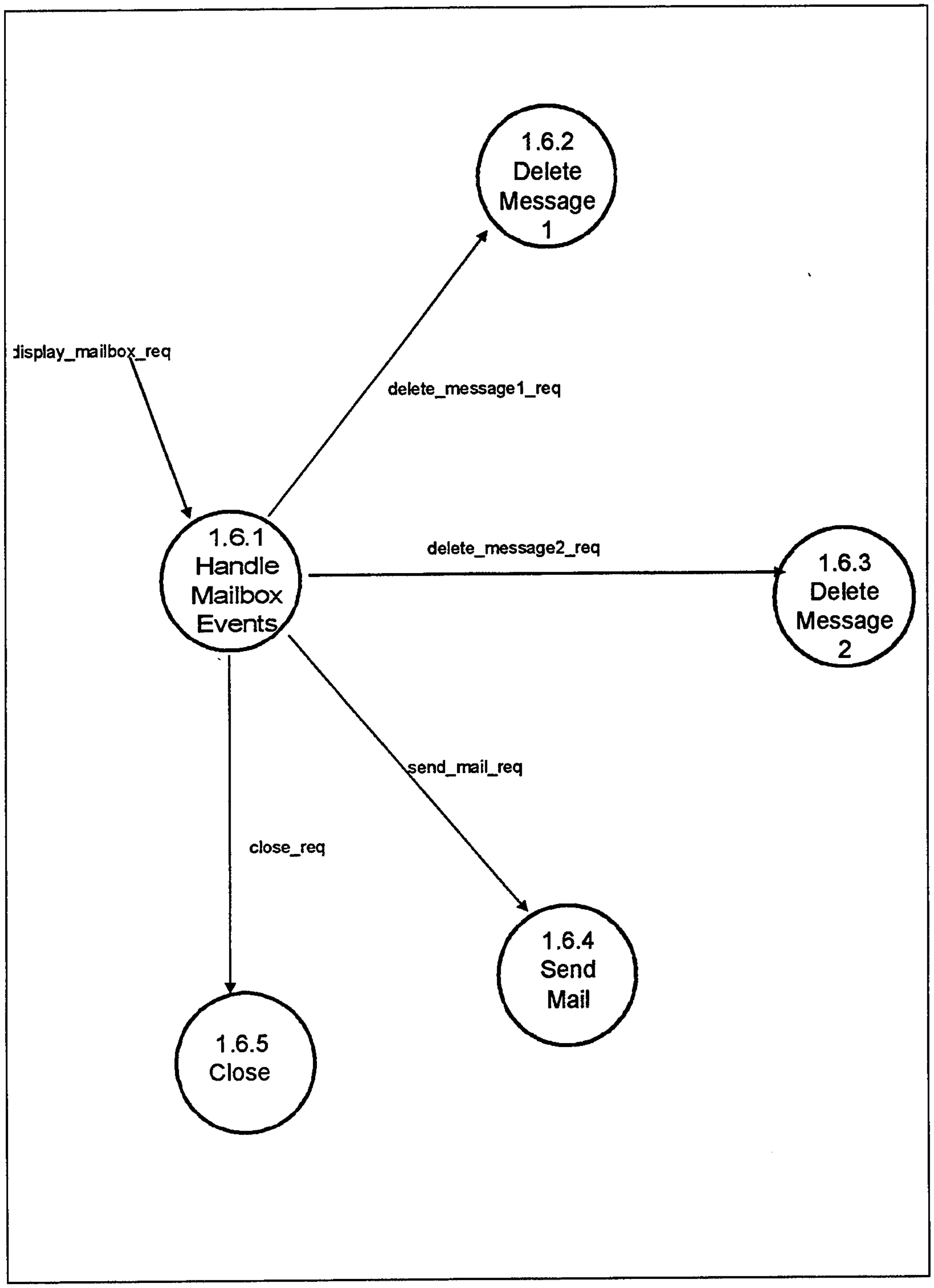
LEVEL 2 (Process 1.3)



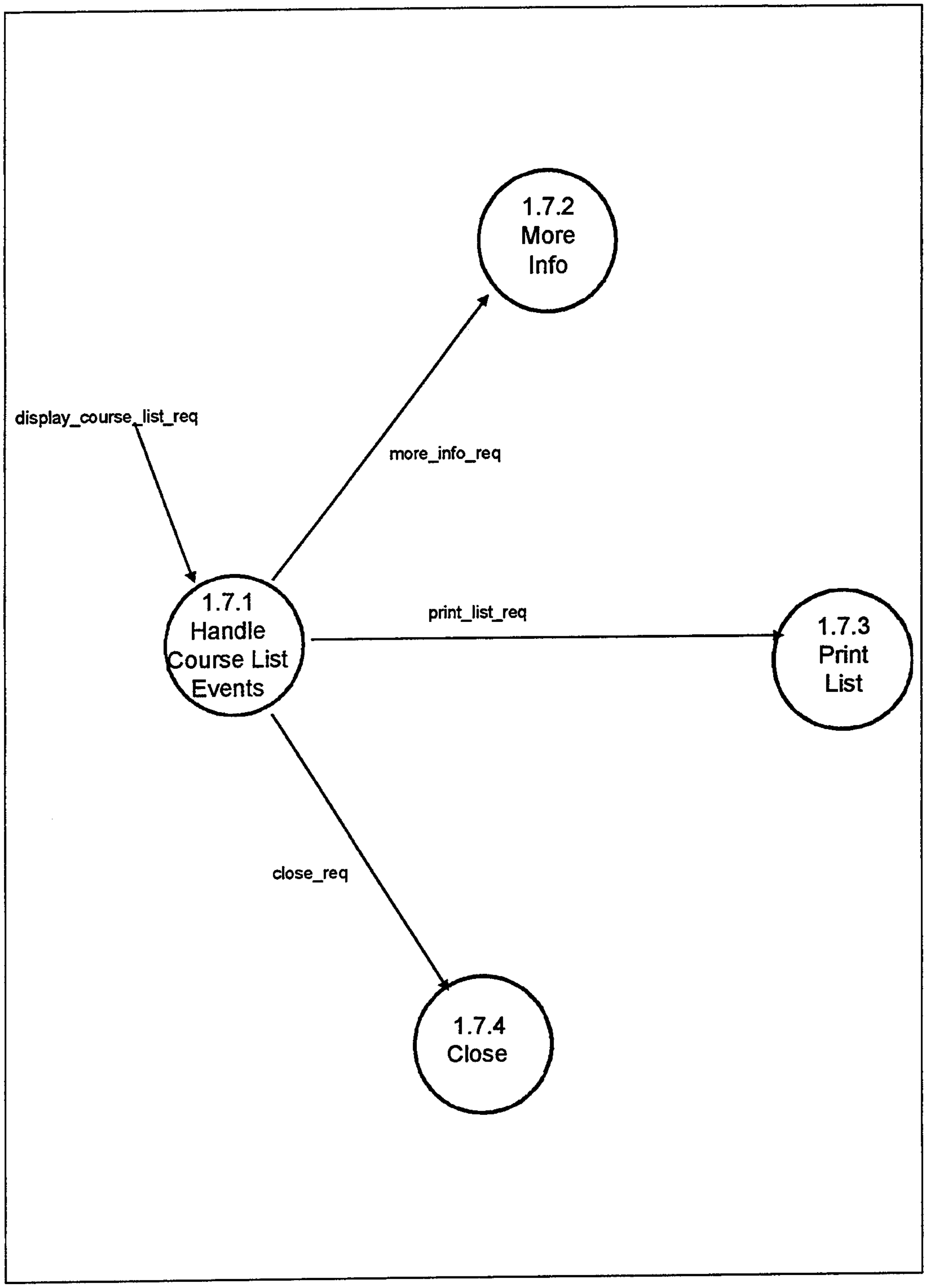
LEVEL 2 (Process 1.4)



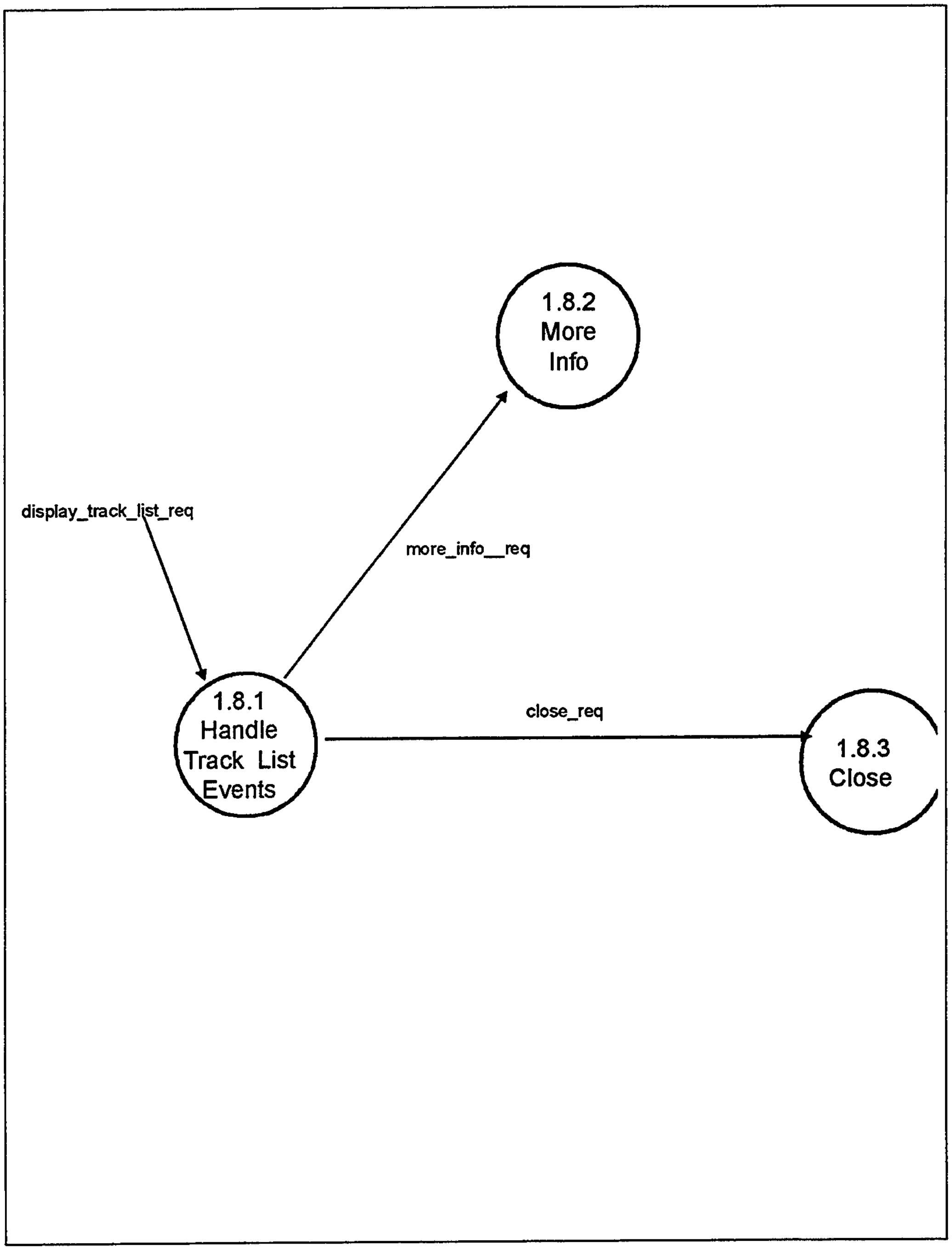
LEVEL 2 (Process 1.5)



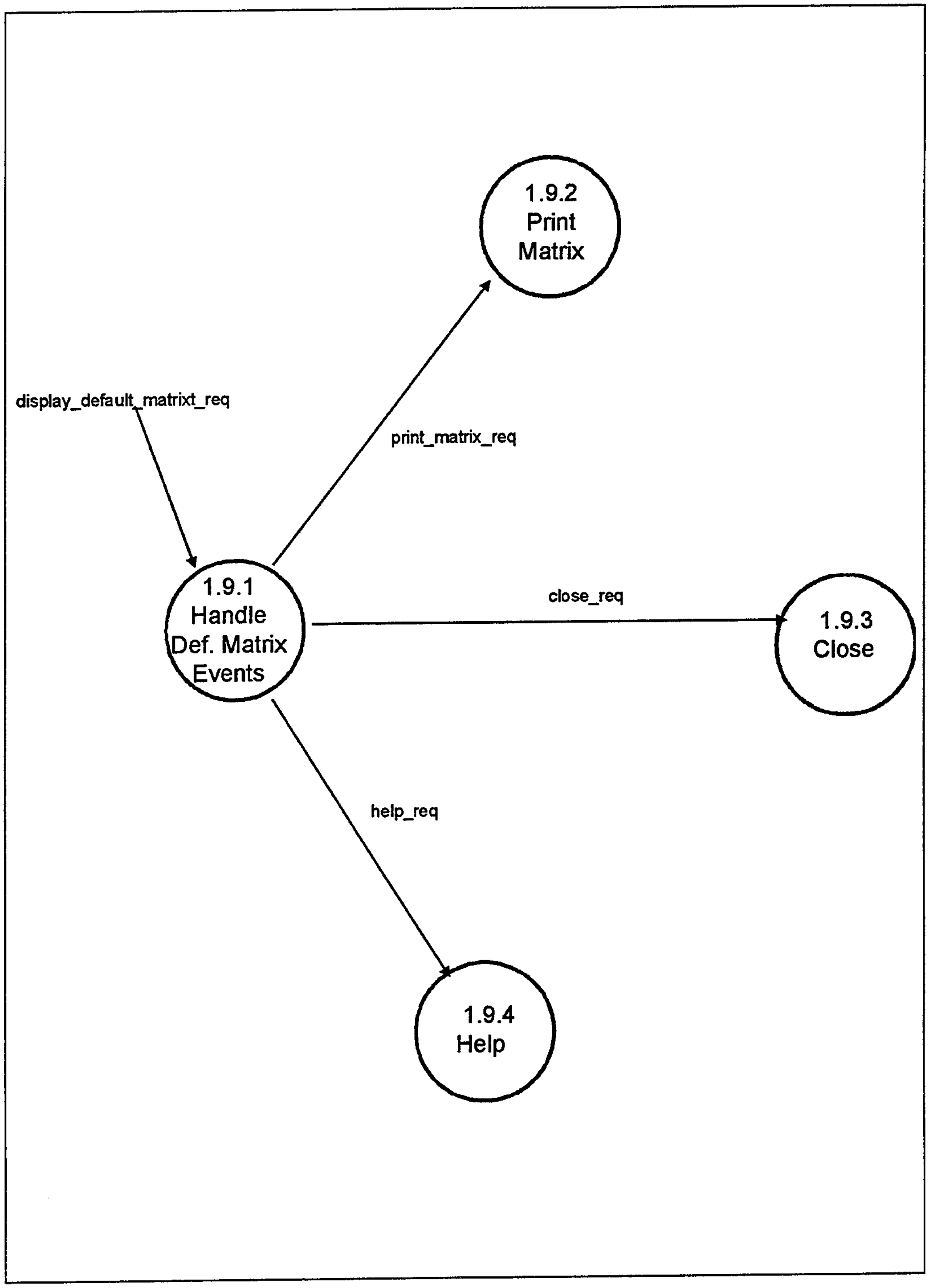
LEVEL 2 (Process 1.6)



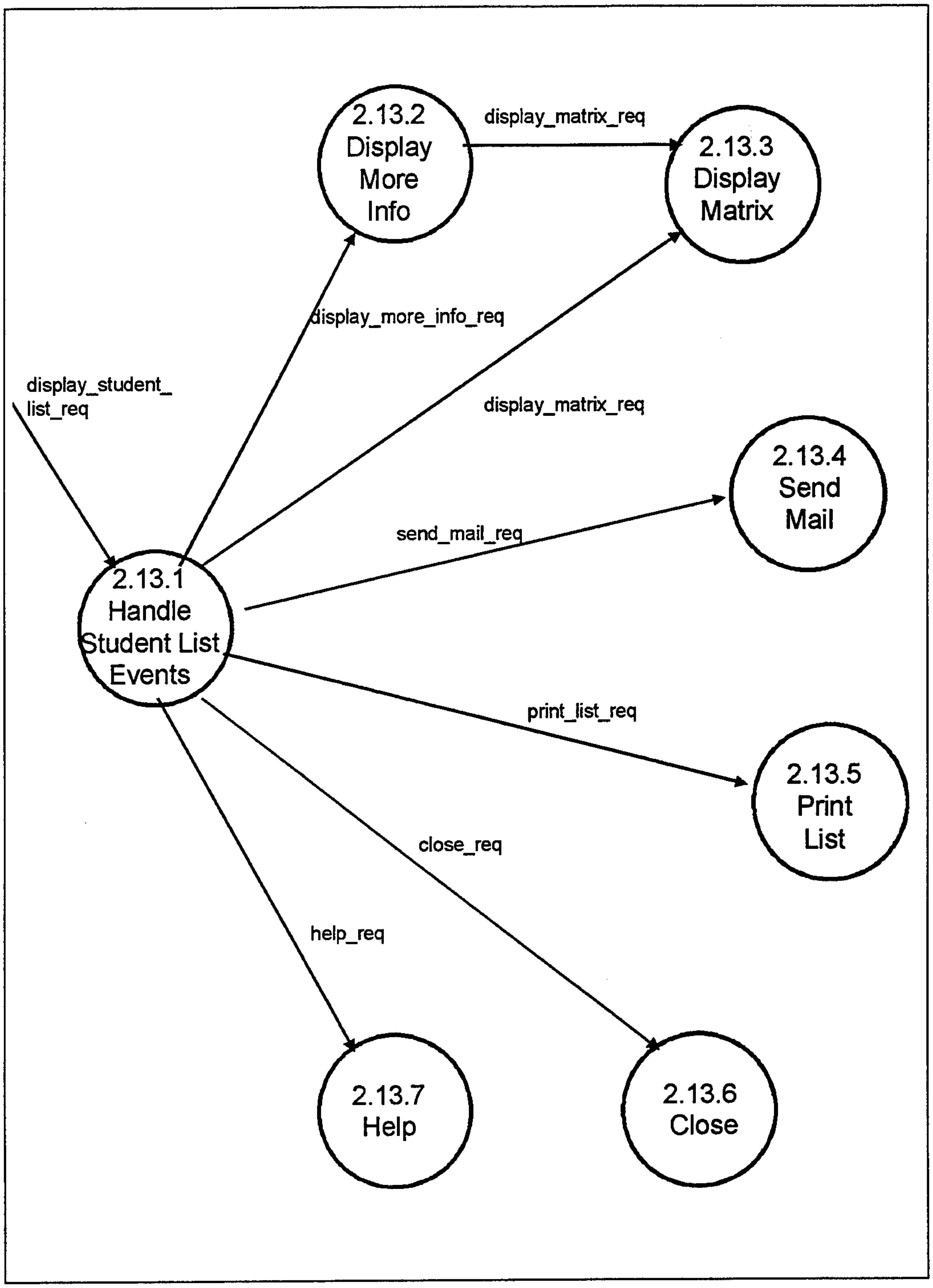
LEVEL 2 (Process 1.7)



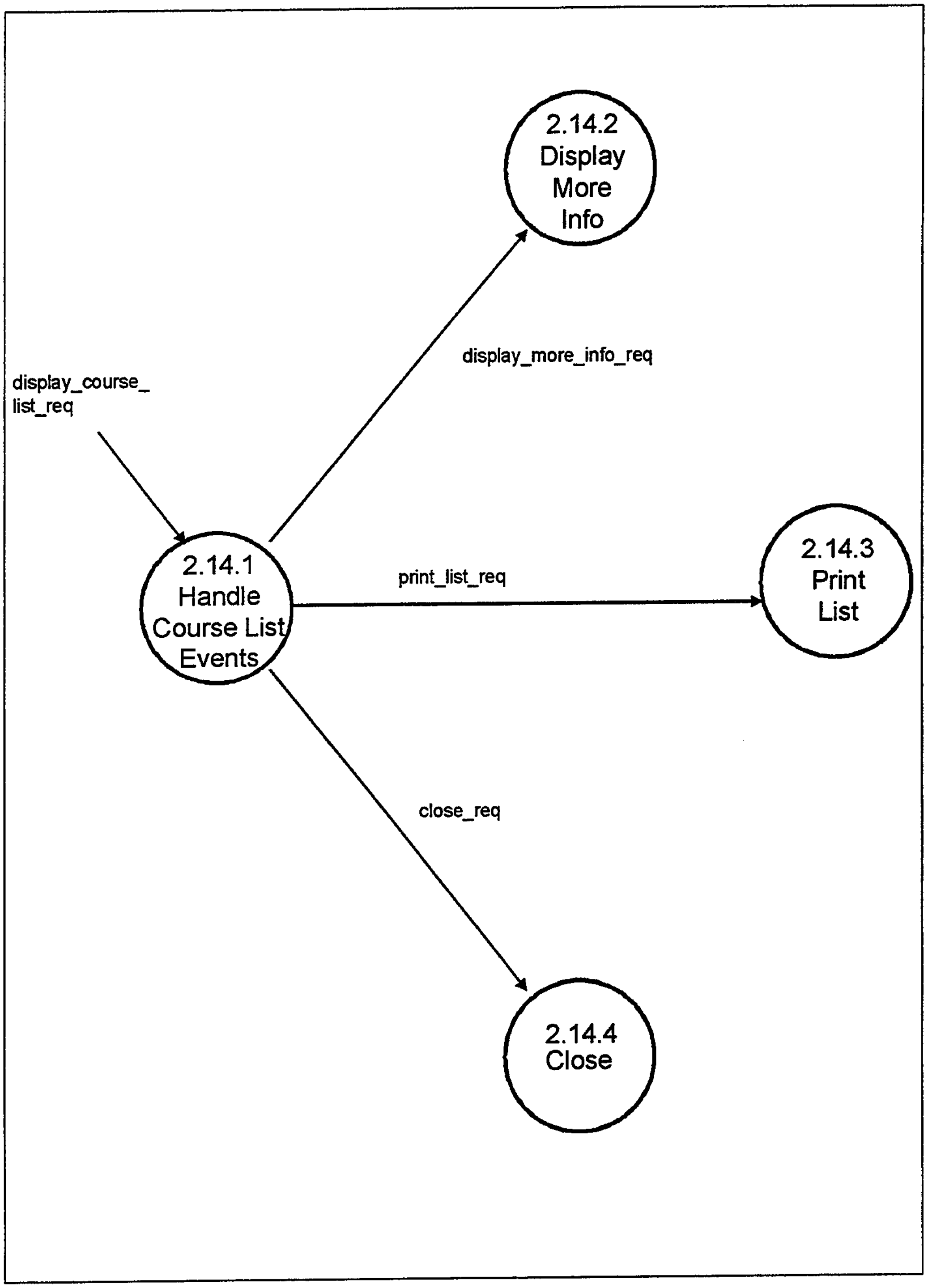
LEVEL 2 (Process 1.8)



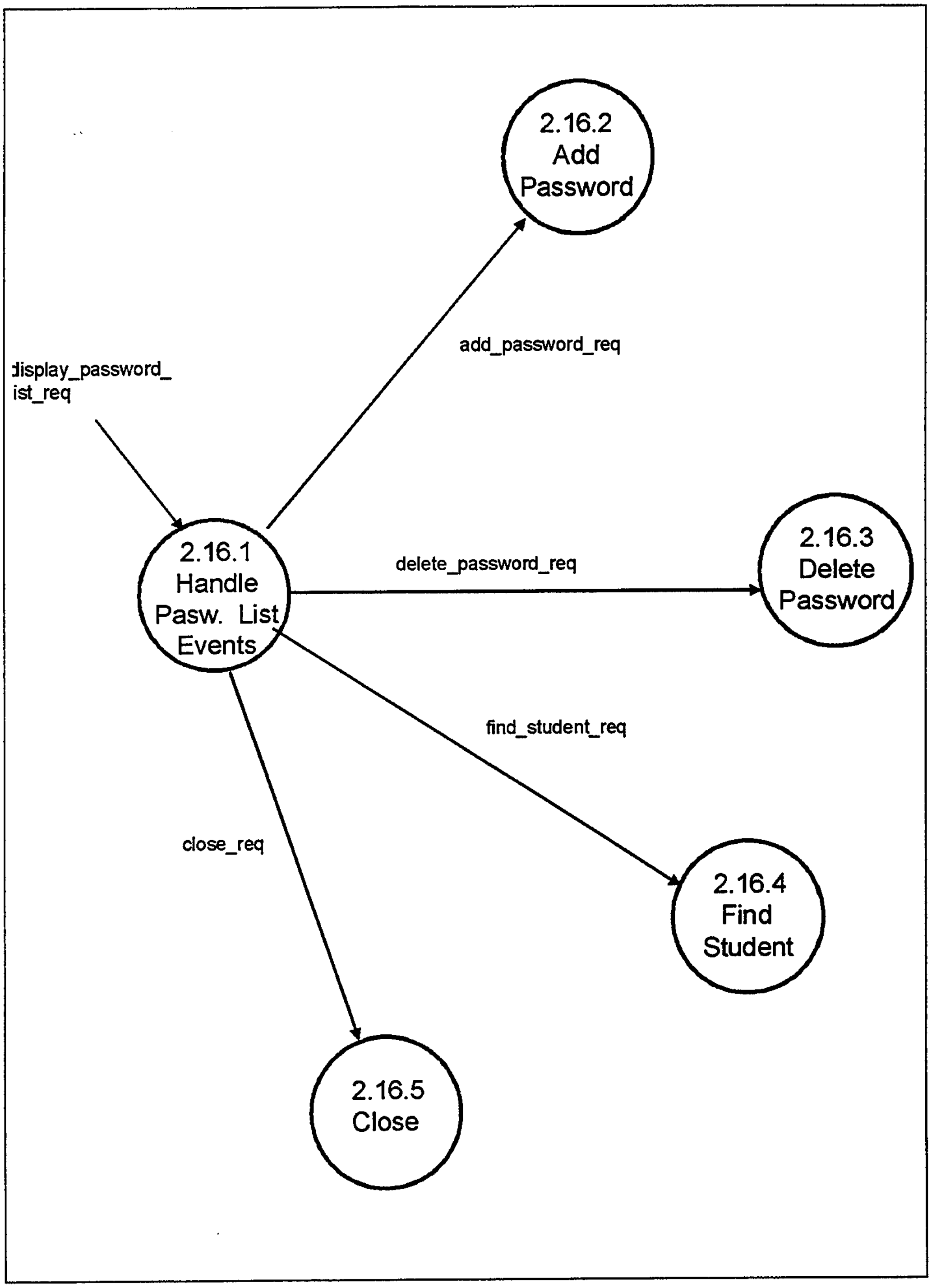
LEVEL 2 (Process 1.9)



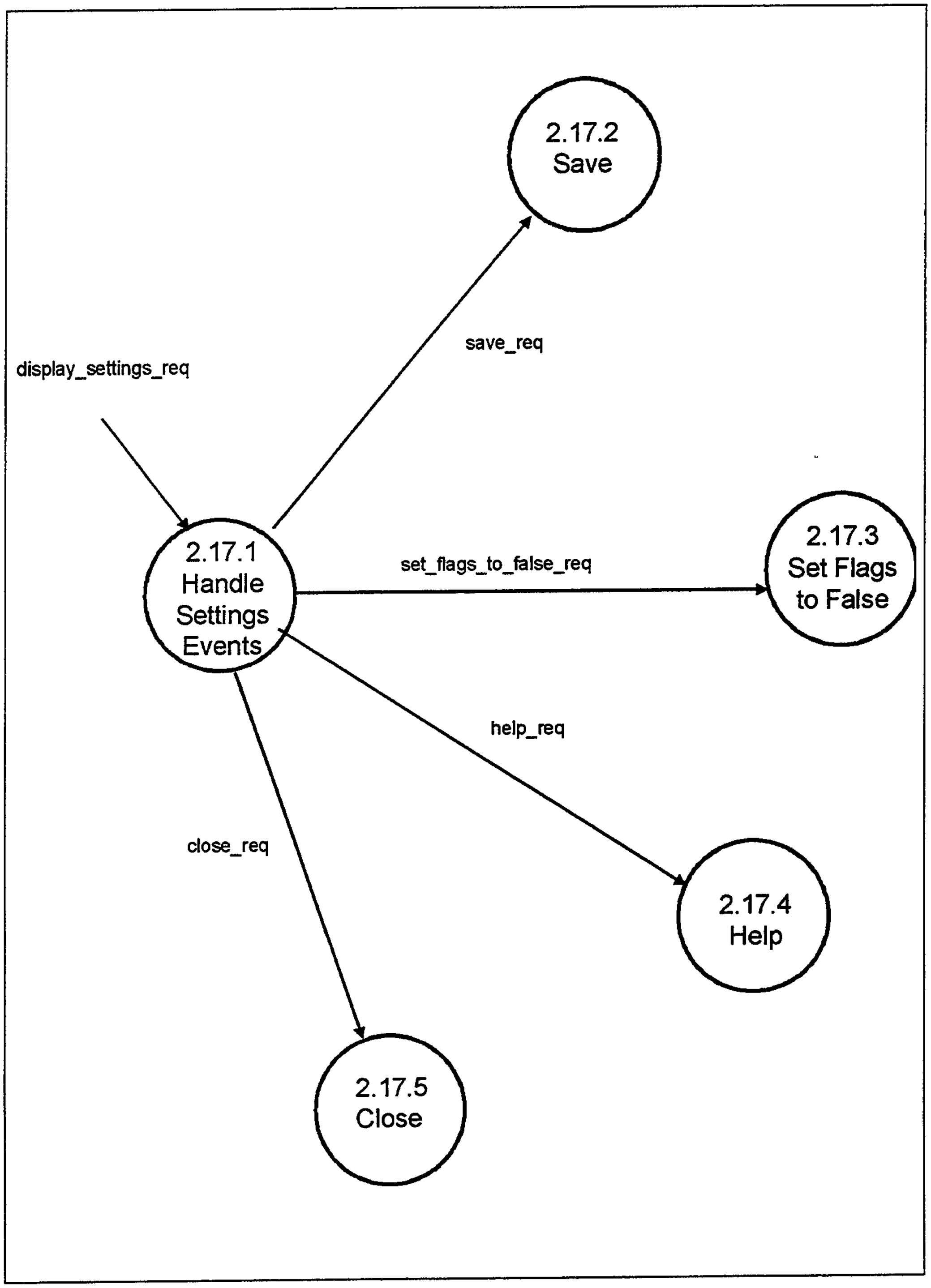
LEVEL 2 (Process 1.13)



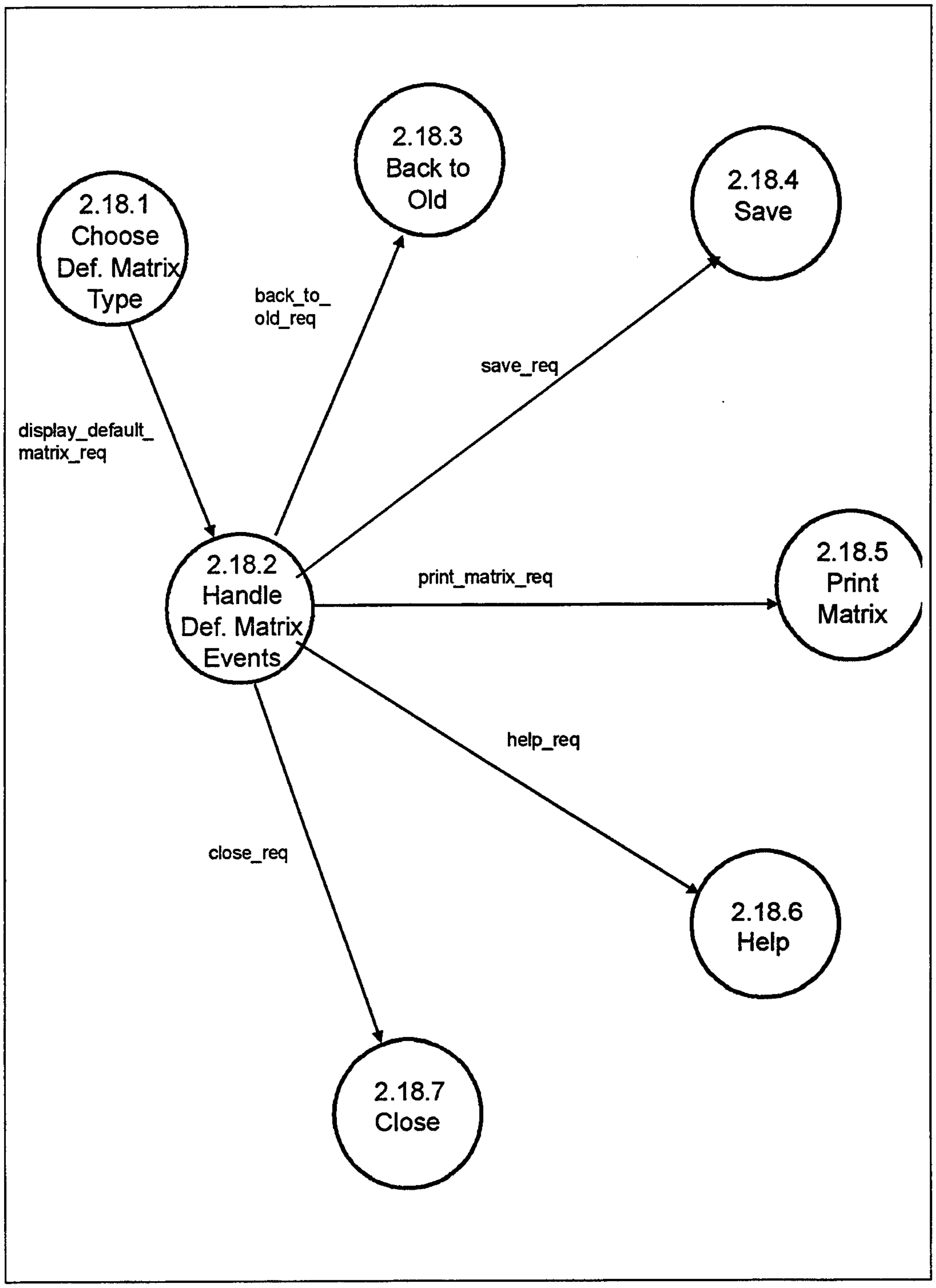
LEVEL 2 (Process 2.14)



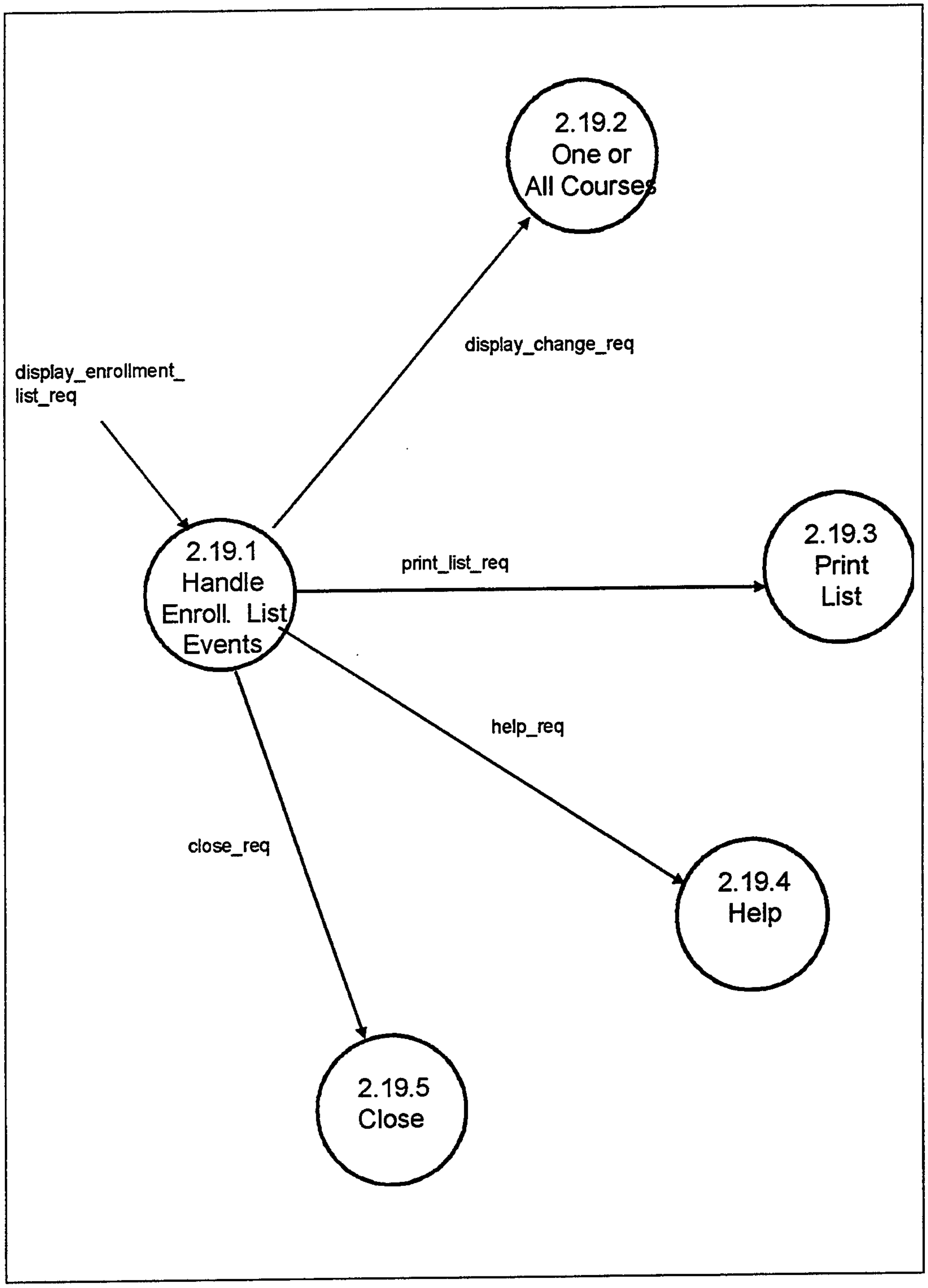
LEVEL 2 (Process 1.16)



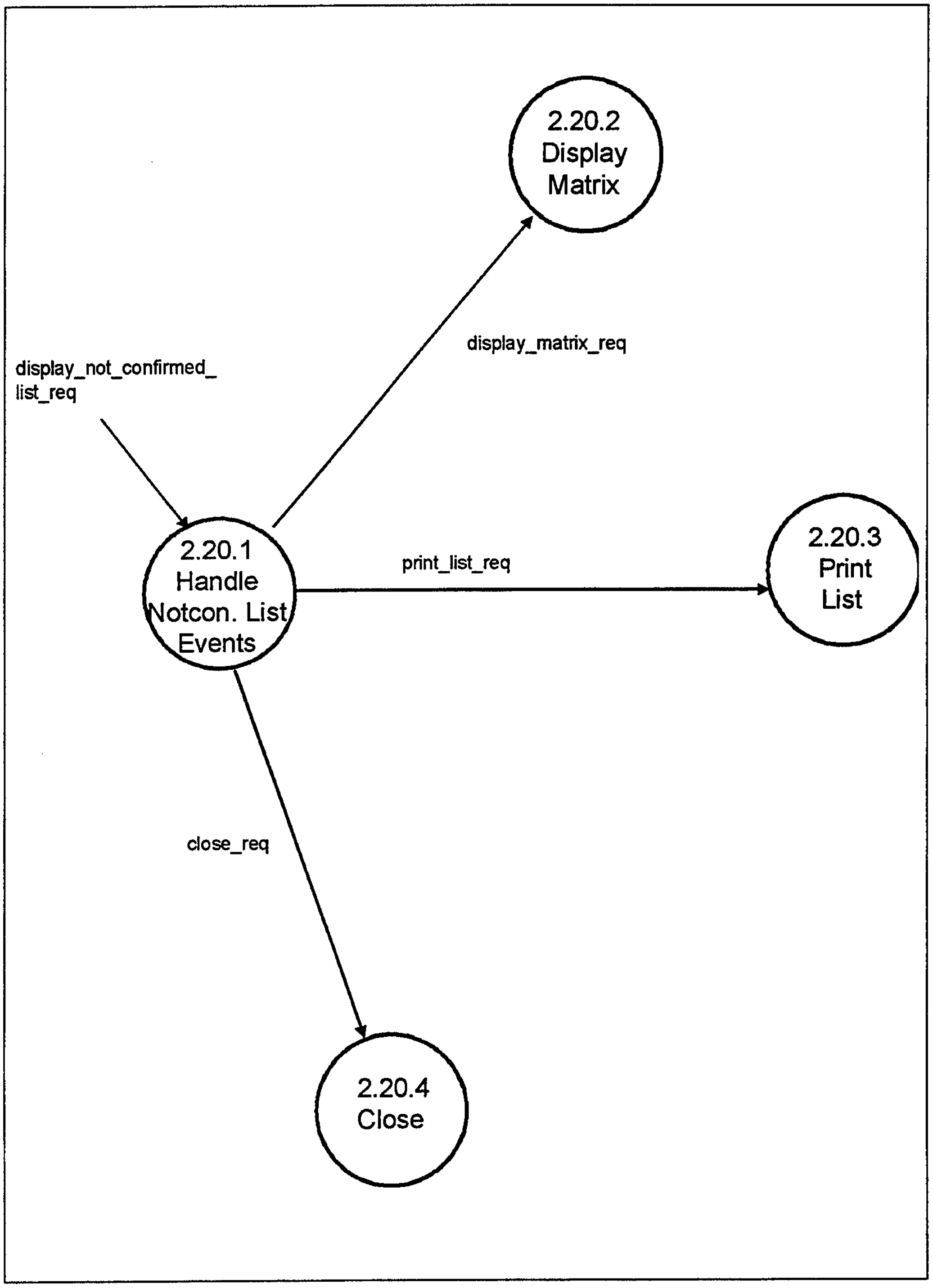
LEVEL 2 (Process 2.17)



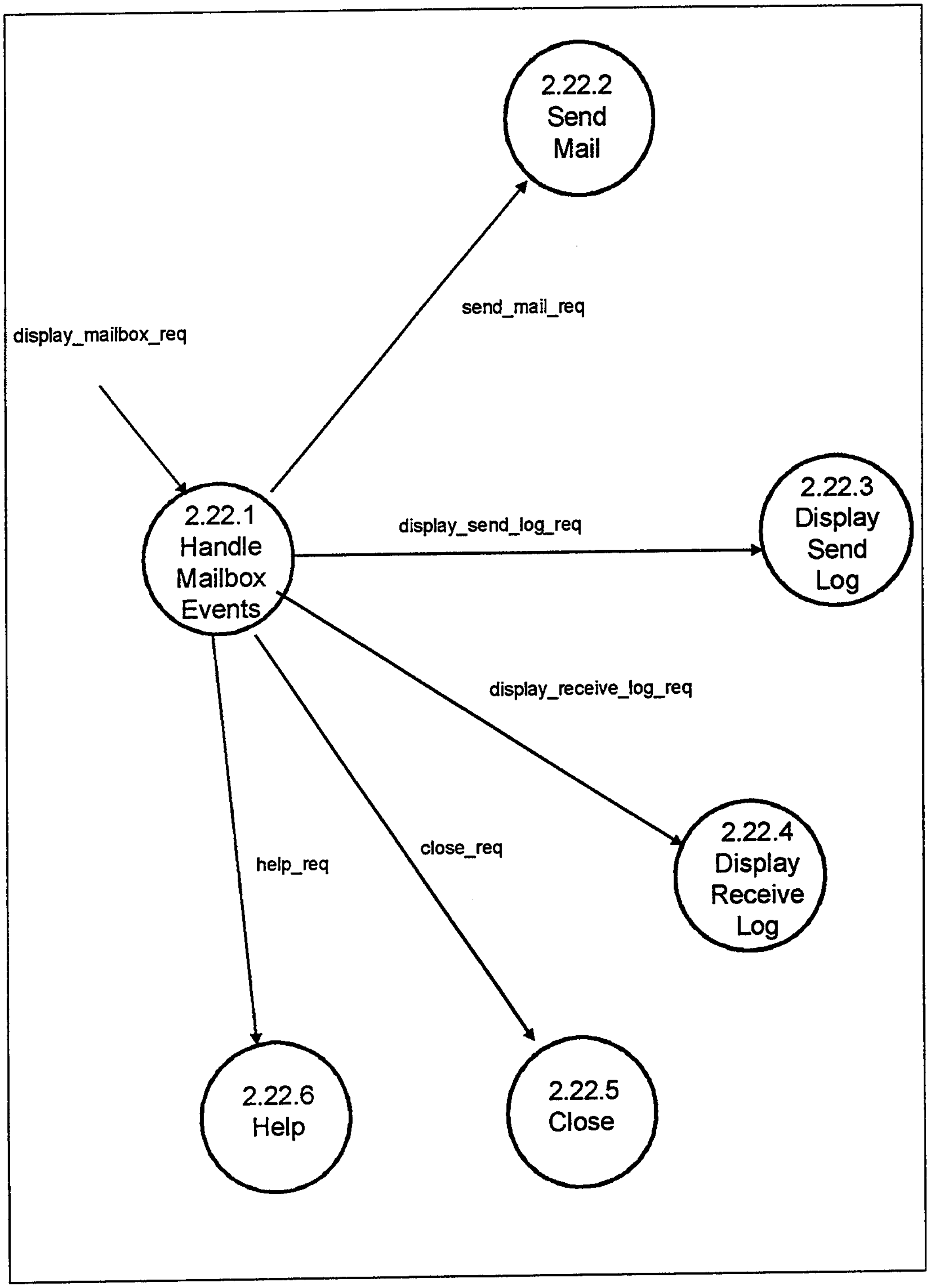
LEVEL 2 (Process 2.18)



LEVEL 2 (Process 2.19)



LEVEL 2 (Process 2.20)



LEVEL 2 (Process 2.22)

APPENDIX B. PROCESS SPECIFICATIONS

Process 1.1 : Check the Password

```
{
    bool loopFlag = TRUE;
    int loopcount = 0;

    while ( loopFlag){
        get(studentSsn);
        get( passwordEntered);
        if ( studentSsn  is in passwordFile){
            get( passwordInFile);
            if ( !(passwordEntered = passwordInFile)){
                print ( " Incorrect Login");
                loopcount = loopcount + 1;
                if ( loopcount = 4){
                    print ("Sorry! You are not allowed to the system");
                    loopFlag = FALSE;
                }
            }
            else
                call process Handle Application Events;
        }
        else{
            print ( " Incorrect Login");
            loopcount = loopcount + 1;
            if ( loopcount = 4)
                print ("Sorry! You are not allowed to the system");
                loopFlag = FALSE;
            }
        }
    }
}
```

Process 1.2 : Handle Application Events

```
{
    display (ApplicationWindow);

    handle events:
        1. Display Matrix
        2. Display Personal Info
        3. Display Privileges
        4. Display Mailbox
        5. Display Course List
        6. Display Track List
        7. Display Default Matrix
        8. Search Course
        9. Change Password
        10. Help
        11. Quit Application

    get  ( choice);
    switch ( choice ){
    case  Display Matrix:
        call process Display Matrix;
    case  Display Personal Info:
        call process Display Personal Info;
    case Display Privileges:
        call process Display Privileges;
    case Display Mailbox:
        call process Display Mailbox;
    case  Display Course List:
        call process Display Course List;
    case  Display Track List:
        call process Track List;
    case  Default Matrix:
        call process Default Matrix;
    case  Search Course:
        call process Search Course;
    case  Change Password:
        call process Change Password;
    case  Help:
        call process Help;
    case  Quit Application:
        call process Quit Application;
    }
}
```

Process 1.3.1 : Handle Matrix Events

```
{
    display (Student's Matrix in Student Matrix Dialog Box);

    handle events:
        1. Select Track
        2. Change Track
        3. Confirm Next Quarter
        4. Deny Next Quarter
        5. Save Matrix
        6. Back To Old
        7. Display Privileges
        8. Display Mailbox
        9. Display Track List
        10. Display Course List
        11. Display Default Matrix
        12. Change Password
        13. Help
        14. Close

    get  ( choice);
    switch ( choice ){
    case  Select Track:
        call process Select Track;
    case  Change Track:
        call process Change Track;
    case Confirm Next Quarter:
        call process Confirm Next Quarter;
    case Deny Next Quarter:
        call process Deny Next Quarter;
    case Save Matrix:
        call process Save Matrix;
    case  Back to Old:
        call process Back to Old;
    case  Display Privileges:
        call process Display Privileges;
    case  Display Mailbox:
        call process Display Mailbox;
    case  Display Track List:
        call process Display Track List;
    case  Display Course List:
        call processDisplay Course List;
    case  Change Password:
        call process Change Password;
    case  Help:
        call process Help;
    case  Close:
        close Student Matix Dialog Box;
    }
}
```

Process 1.3.2 : Select Track

```
{
    if ( a track is already chosen)
        print ( " You already have a track.");
    else{
        open Select Track Dialog Box;
        get ( trackSelected);
        change ( oldTrack to trackSelected);
        write ( track requirements of trackSelected to Student's Matrix);
    }
}
```

Process 1.3.3 : ChangeTrack

```
{
    if ( a track is not chosen before)
        print ( " You do not have a track to change.");
    else{
        open ChangeTrack Dialog Box;
        get ( trackSelected);
        change ( oldTrack to trackSelected);
    }
}
```

Process 1.3.4 : Confirm Next Quarter

```
{
    if ( student can make a confirmation)
        set  Confirmation Flag to TRUE;
}
```

Process 1.3.5 : Deny Next Quarter

```
{
    if ( student can make a confirmation)
        set  Confirmation Flag to FALSE;
}
```

Process 1.3.6 : Save Matrix

```
{
    if ( ( an unrepeatable course is duplicated) OR ( an invalid course is entered) OR
        ( a course is not offered ) OR ( default matrix requirements are not met AND
          no default matrix privilege is given) OR ( track requirements are not met
         AND no track requirements privilege is given) OR ( maximum hour limit is
         exceeded  AND no maximum hour limit is given) )
        print Error Message;
    else
        save Student's Matrix;
}
```

Process 1.3.7 : Back to Old

```
{
    display ( Original Matrix when the Dialog Box was first displayed);
}
```

Process 1.3.8: Close

```
{
    if ( there are unsaved changes){
        print ( "There are unsaved changes. Save ?");
        if ( YES)
            save Student's Matrix;
        else if ( NO){
            not save Student's Matrix;
            close Student Matrix Dialog Box;
        }
        else if ( CANCEL){
            not save Student's Matrix;
            not close Student Matrix Dialog Box;
        }
    }
}
```

Process 1.3.9 : Help

```
{
    display (Help topic for Student Matrix);
}
```

Process 1.4.1 : Handle Info Events

```
{
    display (Student Personal Information in Student Info Dialog Box);

    handle events:
        1. Close
        2. Help

    get ( choice);
    switch ( choice ){
    case  Close:
        call process Close;
    case  Help:
        call process Help;
}
```

Process 1.4.2: Close

```
{
    close Student Info Dialog Box;
}
```

Process 1.4.3 : Help

```
{
    display (Help topic for Student Info);
}
```

Process 1.5.1 : Handle Privileges Events

```
{
    display (Privileges in Privileges Dialog Box);

    handle events:
        1. Close
        2. Help

    get ( choice);
    switch ( choice ){
    case  Close:
        call process Close;
    case  Help:
        call process Help;
}
```

Process 1.5.2: Close

```
{
    close Privileges Dialog Box;
}
```

Process 1.5.3 : Help

```
{
    display (Help topic for Privileges);
}
```

Process 1.6.1 : Handle Mailbox Events

```
{
    display (Mailbox in Mailbox Dialog Box);

    handle events:
        1. Delete Message1
        2. Delete Message2
        3. Send Mail
        4. Close

    get ( choice);
    switch ( choice ){
    case Delete Message1:
        call process Delete Message1;
    case Delete Message2:
        call process Delete Message2;
    case Send Mail:
        call process Send Mail;
    case Close:
        call process Close;
    }
}
```

Process 1.6.2 : Delete Message1

```
{
    delete Message1 from Student's record;
    delete Message1 from Mailbox Dialog Box;
}
```

Process 1.6.3 : Delete Message2

```
{
    delete Message2 from Student's record;
    delete Message2 from Mailbox Dialog Box;
}
```

Process 1.6.4 : Send Mail

```
{
    display (Send Mail Dialog Box);
    get ( writttenMessage);
    send ( written Message )  to ( Curricular Officer's Mailbox);
}
```

Process 1.6.5: Close

```
{
    close Mailbox Dialog Box;
}
```

Process 1.7.1: Handle Course List Events

```
{
    display  (course list in Course List Dialog Box);

    handle events:
        1. More Info
        2. Print List
        3. Close

    get ( choice);
    switch ( choice ){
    case  More Info:
        call process More Info;
    case  Print List:
        call process Print List;
    case Close:
        call process Close;
    }
}
```

Process 1.7.2: More Info

```
{
    display ( selected course in Course Info Dialog Box);

    handle events:
        1. Go to First Course
        2. Jump Back
        3. Go to Previous Course
        4. Go to Next Course
        5. Jump Forward
        6. Go to Last Course
        7. Close

    get ( choice);
    switch ( choice ){
    case Go to First Course:
        display First Course;
    case Jump Back:
        display Course 5 steps back;
    case Go to Previous Course:
        display Previous Course;
    case Go to Next Course:
        display Next Course;
    case Jump Forward:
        display Course 5 steps ahead;
    case Go to Last Course:
        display Last Course;
    case Close:
        close Course Info Dialog Box;
    }
}
```

Process 1.7.3: Print List

```
{
    print list of Courses in the Database;
}
```

Process 1.7.4: Close

```
{
    close Course List Dialog Box;
}
```

Process 1.8.1: Handle Track List Events

```
{
    display  (tracklist in Track List Dialog Box);

    handle events:
        1. More Info
        2. Close

    get ( choice);
    switch ( choice ){
    case  More Info:
        call process More Info;
    case  Close:
        call process Close;
    }
}
```

Process 1.8.2: More Info

```
{
    display ( selected track in Track Info Dialog Box);

    handle events:
        1. Go to First Track
        2. Go to Previous Track
        3. Go to Next Track
        4. Go to Last Track
        5. Close

    get ( choice);
    switch ( choice ){
    case  Go to First Track:
        display (First Track);
    case  Go to Previous Track:
        display (Previous Track);
    case  Go to Next Track:
        display (Next Track);
    case  Go to Last Track:
        display (Last Track);
    case  Help:
        display (Help Topic for Track Info);
    case  Close:
        close Track Info Dialog Box;
    }
}
```

Process 1.8.3: Close

```
{
    close Track List Dialog Box;
}
```

Process 1.9.1: Handle Default Matrix Events

```
{
    display  (default matrix for the student in Default Matrix Dialog Box);

    handle events:
        1. Print Matrix
        2. Close
        3. Help

    get ( choice);
    switch ( choice ){
    case  Print Matrix;
```

```
        call process Print Matrix;
    case Close;
        call process Close;
    case Help;
        call process Help;
    }
}
```

Process 1.9.2: Print Matrix

```
{
    print currently displayed Default Matrix;
}
```

Process 1.9.3: Close

```
{
    close Default Matrix Dialog Box;
}
```

Process 1.9.4: Help

```
{
    display (Help Topic for Default Matrix);
}
```

Process 1.10: Search Course

```
{
    open Search Course Dialog Box;
    get ( courseName to be searched for);
    if ( found( courseName)){
        open Course Info Dialog Box;
        display (course found);
    }
    else
        print ("Not Found");
}
```

Process 1.11: Change Password

```
{
    open Change Password Dialog Box;
    get ( newPassword);
    get ( newPasswordAgain);
    if ( newPassword == newPasswordAgain)
        replace old password with newPassword;
    else
        print ( "different passwords entered");
}
```

Process 1.12: Help

```
{
    display (Help Contents Window);
}
```

Process 1.13: Quit Application

```
{
    quit Automated Course Enrollment System Application;
}
```

Process 2.1 : Check the Password

```
{
    bool loopFlag = TRUE;
    int loopcount = 0;

    while ( loopFlag){
        get (studentSsn);
        get ( passwordEntered);
        if ( studentSsn  == "CURRIC"){
            get( authorizedPassword) from Settings File;
            if ( !(passwordEntered == authorizedPassword)){
                print ( " Incorrect Login");
                loopcount = loopcount + 1;
                if ( loopcount = 4){
                    print ("Sorry! You are not allowed to the system");
                    loopFlag = FALSE;
                }
            }
            else
                call process Handle Application Events;
        }
        else{
            print ( " Incorrect Login");
            loopcount = loopcount + 1;
            if ( loopcount = 4)
                print ("Sorry! You are not allowed to the system");
                loopFlag = FALSE;
            }
        }
    }
}
```

Process 2.2 : Handle Application Events

```
{
    display (ApplicationWindow);

    handle events:
        1. Add Student
        2. Add Course
        3. Add Track
        4. Add Password
        5. Delete Student
        6. Delete Course
        7. Delete Track
        8. Search Student By Name
        9. Search Student By SSN
        10. Search Course
        11. Display Student List
        12. Display Course List
        13. Display Track List
        14. Display Password List
    15. Display Settings
    16. Display Default Matrix
    17. Display Enrollment List
    18. Display Not Confirm List
    19. Generate Enrollment List
    20. Display MailBox
    21. Quit Application

get ( choice);
switch ( choice ){
case Add Student:
    call process Add Student;
case Add Course:
    call process Add Course;
case Add Track:
    all process Add Track;
case Add Password:
    call process Add Password;
case Delete Student:
    call process Delete Student;
case Delete Course;
    call process Delete Course;
case Delete Track;
    call process Delete Track;
case Search Student By Name:
    call process Search Student By Name;
case Search Student By SSN:
    call process Search Student By SSN;
case Search Course:
    call process Search Course;
case Display Student List:
    call process Display Student List;
case Display Course List;
    call process Display Course List;
case Display Track List:
    call process Display Track List;
case Display Password List;
    call process Display Password List;
case Display Settings:
    call process Display Settings;
case Display Default Matrix;
    call process Display Default Matrix;
    case  Display Enrollment List:
        call process Display Enrollment List;
    case  Display Not Confirm List;
        call process Display Not Confirm List;
    case  Generate Enrollment List:
        call process Enrolment List;
    case  Display Mailbox;
        call process Display Mailbox;
    case  Quit Application:
        loopFlag = FALSE;
    }
}
```

Process 2.3: Add Student

```
{
    open Add Student Dialog Box;
    get (studentInformation);
    if ( valid( studentInformation))
        add a new student to the students database with studentInformation;
    else
        print ( "Input Error");
}
```

Process 2.4: Add Course

```
{
    open Add Course Dialog Box;
    get (courseInformation);
    if ( valid( courseInformation))
        add a new course to the courses database with courseInformation;
    else
        print ( "Input Error");
}
```

Process 2.5: Add Track

```
{
    open Add Track Dialog Box;
    get (trackInformation);
    if ( valid( trackInformation))
        add a new track to the tracks database with trackInformation;
    else
        print ( "Input Error");
}
```

Process 2.6: Add Password

```
{
    open Add Password Dialog Box;
    get (passwordInformation);
    if ( valid( passwordInformation))
        add a new password to the passwords database with passwordInformation;
    else
        print ( "Input Error");
}
```

Process 2.7: Delete Student

```
{
    open Delete Student Dialog Box;
    get ( student) to be deleted;
    if ( student is in database)
        remove student from the students database;
    else
        print ( "Student not found");
}
```

Process 2.8: Delete Course

```
{
    open Delete Course Dialog Box;
    get ( course) to be deleted;
    if ( course is in database)
        remove course from the courses database;
    else
        print ( "Course not found");
}
```

Process 2.9: Delete Track

```
{
    open Delete Track Dialog Box;
    get ( track) to be deleted;
    remove course from the courses database;
}
```

Process 2.10: Search Student By Name

```
{
    open Search Student By Name Dialog Box;
    get ( studentName) to be deleted;
    if ( studentName is in database)
        display (found student in Student Info Dialog Box);
    else
        print ( Student not found");
}
```

Process 2.11: Search Student By SSN

```
{
    open Search Student By SSN Dialog Box;
    get ( studentSSN) to be deleted;
    if ( studentSSN is in database)
        display (found student in Student Info Dialog Box);
    else
        print ( Student not found");
}
```

Process 2.12: Search Course

```
{
    open Search Course Dialog Box;
    get ( courseNo) to be deleted;
    if ( courseNo is in database)
        display (found course in Course Info Dialog Box);
    else
        print ( Course not found");
}
```

Process 2.13.1: Handle Student List Events

```
{
    display (studentlist in Student List Dialog Box);

    handle events:
        1. Display More Info
        2. Display Matrix
        3. Send Mail
        4. Print List
        5. Close
        6. Help

    get  ( choice);
    switch ( choice ){
    case Display More Info:
        call process Display More Info:
    case Display Matrix:
        call process Display Matrix;
    case Send Mail:
        open Send Mail Dialog Box;
    case  Print List:
        print student list displayed;
    case Close:
        close Student List Dialog Box;
    case Help:
        display Help Topic for Student List;
    }
}
```

Process 2.13.2: Display More Info

```
{
    display (selected student's information in Student Info Dialog Box);

    handle events:
        1. Go to First Student
        2. Jump Backward
        3. Go to Previous Student
        4. Jump Forward
    5. Go to Next Student
    6. Go to Last Student
    7. Save
    8. Delete
    9. Matrix
    10. Close
    11. Help

get ( choice);
switch ( choice){
case Go to First Student:
    display (First Student);
case Jump Backward:
    go 5 students back in the database from the currently displayed student.
    display (Student);
case Go to Previous Student:
    display (Previous Student);
case Go to Next Student:
    display (Next Student);
case Jump Forward:
    go 5 students back in the database from the currently displayed student.
    display (Student);
case Go to Last Student
    display (Last Student);
case  Save:
    if ( valid( studentInformation))
        update displayed student in the database  with studentInformation;
    else
        print ( "Input Error");
case Delete:
    remove currently displayed student from the database;
case Matrix:
    display (currently displayed student's matrix in Student matrix Dialog Box);
case Close:
    if ( there are unsaved changes){
        print ( "There are unsaved changes. Save ?");
        if ( YES)
            save Student's Info;
        else if ( NO){
            not save Student's Info;
            close Student Info Dialog Box;
        }
```

```
            else if ( CANCEL){
                not save Student's Info;
                not close Student Info Dialog Box;
            }
        }
    case Help:
        display (Help Topic for Student Info);
    }
}
```

Process 2.13.3: Display Matrix

```
{
    display (selected student's matrix in Student Matrix Dialog Box);

    handle events:
        1. Select Track
        2. Change Track
        3. Confirm Next Quarter
        4. Deny Next Quarter
        5. Save Matrix
        6. Back To Old
        7. Display Privileges
        8. Display Mailbox
        9. Display Track List
        10. Display Course List
        11. Display Default Matrix
        12. Change Password
        13. Help
        14. Close

    get  ( choice);
    switch ( choice ){
    case  Select Track:
        if ( a track is already chosen)
            print ( " You already have a track.");
        else{
            display (Select Track Dialog Box);
            get ( trackSelected);
            change ( oldTrack to trackSelected);
            write ( track requirements of trackSelected to Student's Matrix);
    }
case  Change Track:
    if ( a track is not chosen before)
        print ( " You do not have a track to change.");
    else{
        display (ChangeTrack Dialog Box);
        get ( trackSelected);
        change ( oldTrack to trackSelected);
    }
case Confirm Next Quarter:
    if ( student can make a confirmation)
        set  Confirmation Flag to TRUE;
case Deny Next Quarter:
    if ( student can make a confirmation)
        set  Confirmation Flag to FALSE;
case Save Matrix:
    if ( ( an unrepeatable course is duplicated) OR ( an invalid course is entered)
        OR ( a course is not offered ) )
        print Error Message;
    else
        save Student's Matrix;
case  Back to Old:
    display (Original Matrix when the Dialog Box was first displayed);
case  Display Privileges:
    display (Privileges in Privileges Dialog Box);
case  Display Mailbox:
    display (Mailbox in Mailbox Dialog Box);
case  Display Track List:
    display (Track List in Track List Dialog Box);
case  Display Course List:
    display (Course List in Course List Dialog Box);
case  Change Password:
    open Change Password Dialog Box;
    get ( newPassword);
    get ( newPasswordAgain);
    if ( newPassword == newPasswordAgain)
        replace old password with newPassword;
    else
        print ( "different passwords entered");
case  Help:
    display  (Help Topic for Student Matrix);
case  Close:
```

```
            if ( there are unsaved changes){
                print ( "There are unsaved changes. Save ?");
                if ( YES)
                    save Student's Matrix;
                else if ( NO){
                    not save Student's Matrix;
                    close Student Matrix Dialog Box;
                }
                else if ( CANCEL){
                    not save Student's Matrix;
                    not close Student Matrix Dialog Box;
                }
            }
        }
    }
```

Process 2.13.4: Send mail

```
{
    open Send Mail Dialog Box with the selected student as the recipient;
    get  ( message) to be sent;
    send ( message) to student's mailbox;
}
```

Process 2.13.5: Print List

```
{
    get ( choice) for the format of the print out;
    print displayed Student List according to format selected;
}
```

Process 2.13.6: Close

```
{
    close Student List Dialog Box;
}
```

Process 2.13.7: Help

```
{
    display (Help Topic for Student List);
}
```

Process 2.14.1: Handle Course List Events

```
{
    display  (course list in Course List Dialog Box);

    handle events:
        1. More Info
        2. Print List
        3. Close

    get ( choice);
    switch ( choice ){
    case  More Info:
        call process Display More Info;
    case  Print List:
        call process Print List;
    case  Close:
        call process Close;
    }
}
```

Process 2.14.2: Display More Info

```
{
    display ( selected course in Course Info Dialog Box);

    handle events:
        1. Go to First Course
        2. Jump Back
        3. Go to Previous Course
        4. Go to Next Course
        5. Jump Forward
        6. Go to Last Course
        7. Save
        8. Delete
        9. Close
    get ( choice);
    switch ( choice ){
    case  Go to First Course:
        display (First Course);
    case  Jump Back:
        display (Course 5 steps back);
    case  Go to Previous Course:
        display (Previous Course);
    case  Go to Next Course:
        display (Next Course);
    case  Jump Forward:
        display (Course 5 steps ahead);
    case  Go to Last Course:
        display (Last Course);
    case  Save:
        get coursetInformation;
        if ( valid( courseInformation))
            update the currently displayed course with courseInformation;
        else
            print ( error message);
    case  Delete:
        remove currently displayed course from the database;
    case  Close:
        if ( there are unsaved changes){
            print ( "There are unsaved changes. Save ?");
            if ( YES)
                save Course Info;
            else if ( NO){
                do not save Course Info;
                close Course Info Dialog Box;
            }
            else if ( CANCEL){
                do not save Course Info;
                do not close Course Info Dialog Box;
            }
        }
    }
}
```

Process 2.14.3: Print List

```
{
    print list of Courses in the Database;
}
```

Process 2.14.4: Close

```
{
    close Course List Dialog Box;
}
```

Process 2.15: Display Track List

```
{
    display (list of Tracks in Track List Dialog Box);

    handle events:
        1. More Info
        2. Close

    get ( choice);
    switch ( choice ){
    case  More Info:
        display ( selected track in Track Info Dialog Box);

        handle events:
            1. Go to First Track
            2. Go to Previous Track
            3. Go to Next Track
            4. Go to Last Track
            5. Add Required Course
            6. Delete Required Course
            7. More Info for Required Course
            8. Add Track Elective
            9. Delete Track Elective
            10. More Info for Track Elective
            11. Delete Track
        12. Save Track
        13. Help
        14. Close

    get ( choice);
    switch ( choice ){
    case  Go to First Track:
        display (First Track);
    case  Go to Previous Track:
        display (Previous Track);
    case  Go to Next Track:
        display (Next Track);
    case  Go to Last Track:
        display (Last Track);
    case  Add Required Course:
        open Add Required Course Dialog Box;
        get   a Required Course;
        put Required Course in Required Courses List Box;
    case  Delete Required Course:
        delete selected Required Course from the Required Courses List Box;
    case  More Info for Required Course:
        open Course Info Dialog Box;
        display (selected Required Course);
    case  Add Track Elective:
        open Add Track Elective Dialog Box;
        get   a Track Elective;
        put Track Elective inTrack Electives List Box;
    case  Delete Track Elective:
        delete selected Track Elective from the Track Electives List Box;
    case  More Info for Track Elective:
        open Course Info Dialog Box;
        display (selected Track Elective);
    case  Delete Track:
        remove currently displayed track from the tracks database;
    case  Save Track:
        save currently displayed track;
    case  Help:
        display (Help Topic for Track Info Dialog Box);
    case  Close:
        close Track Info Dialog Box;
    }
case  Close:
        close Track List Dialog Box;
    }
}
```

Process 2.16.1: Handle Password List Events

```
{
    display ( list of passwords in Password List Dialog Box);

    handle events:
        1. Add Password
        2. Delete Password
        3. Find Student
        4. Close

    get ( choice);
    switch ( choice ){
    case  Add Password:
        call process Add Password;
    case  Delete Password:
        call process Delete Password;
    case  Find Student:
        call process Find Student;
    case  Close:
        call process Close;
    }
```

Process 2.16.2: Add Password

```
{
    open Add Password Dialog Box;
    get studentSSN;
    get  password;
    if ( (studentSSN is not in Student Database) OR (studentSSN already has a
        password)
        OR ( password is already in database))
        print ( Error Message);
    else
        add password to the database for studentSSN;
}
```

Process 2.16.3: Delete Password

```
{
    remove selected password from the database;
}
```

Process 2.16.4: Find Student

```
{
    display (selected student in Student Info Dialog Box);
}
```

Process 2.16.5: Close

```
{
    close Password List Dialog Box;
}
```

Process 2.17.1: Handle Settings Events

```
{
    display ( settings in Settings Dialog Box);

    handle events:
        1. Save
        2. Set Flags to False
        3. Help
        4. Close

    get ( choice);
    switch ( choice ){
    case  Save:
        call process Save;
    case  Delete Password:
        call process Delete Password;
    case  Find Student:
        call process Find Student;
    case  Close:
        call process Close;
    }
}
```

Process 2.17.2: Save

```
{
    if ( (entered courses are valid) AND ( required information is entered))
        save Settings Information;
    else
        print ( Error Message);
}
```

Process 2.17.3: Set Flags to False

```
{
    set ( Confirmation Flags for every student to FALSE);
}
```

Process 2.17.4: Help

```
{
    display (Help Topic for Settings);
}
```

Process 2.17.5: Close

```
{
    if ( there are unsaved changes){
        print ( "There are unsaved changes. Save ?");
        if ( YES)
            save Settings;
        else if ( NO){
            do not save Settings;
            close Settings Dialog Box;
        }
        else if ( CANCEL){
            do not save Settings;
            do not close Settings Dialog Box;
        }
    }
}
```

Process 2.18.1: Choose Default Matrix Type

```
{
    get ( default matrix type);
    call process Handle Default Matrix Events;
}
```

Process 2.18.2: Handle Default Matrix Events

```
{
    display ( default matrix  in Default Matrix Dialog Box);

    handle events:
        1. Back to Old
        2. Save
        3. Print Matrix
        4. Help
        5. Close

    get ( choice);
    switch ( choice ){
    case Back to Old:
        call process Back to Old;
    case Save:
        call process Save;
    case  Print Matrix:
        call process Print Matrix;
    case  Help:
        call process Help;
    case  Close:
        call process Close;
    }
```

Process 2.18.3: Back to Old

```
{
    if ( (entered courses are valid) AND  ( required information is entered))
        save Settings Information;
    else
        print ( Error Message);
}
```

Process 2.18.4: Save

```
{
    set (  Confirmation Flags for every student to FALSE);
}
```

Process 2.18.5: Print Matrix

```
{
    print ( currently displayed default matrix);
}
```

Process 2.18.6: Help

```
{
    display  (Help Topic for Default Matrix);
}
```

Process 2.18.7: Close

```
{
    if ( there are unsaved changes){
        print ( "There are unsaved changes. Save ?");
        if ( YES)
            save Default Matrix;
        else if ( NO){
            do not save Default Matrix;
            close Default Matrix Dialog Box;
        }
        else if ( CANCEL){
            do not save Default Matrix;
            do not close Default Matrix Dialog Box;
        }
    }
}
```

Process 2.19.1: Handle Enrollment List Events

```
{
    display ( Enrollment List  in Enrollment List Dialog Box);

    handle events:
        1. One or All Courses
        2. Print List
        3. Help
        4. Close

    get ( choice);
    switch ( choice ){
    case One or All Courses:
        call process One or All Courses;
    case Print List:
        call process Print List;
    case  Help:
        call process Help;
    case  Close:
        call process Close;
    }
}
```

Process 2.19.2: One or All Courses

```
{
    if ( One Course is selected){
        get ( CourseNo);
        display ( students enrolled for CourseNo);
    }
    else
        display ( all students enrolled for any course);
}
```

Process 2.19.3: Print List

```
{
    print (  Enrollment List);
}
```

Process 2.19.4: Help

```
{
    display  (HelpTopic for enrollment List);
}
```

Process 2.19.5: Close

```
{
    close  Enrollment List Dialog Box;
}
```

Process 2.20.1: Handle Not Confirmed List Events

```
{
    display ( list of students who did not confirm their next quarters' schedule
                    in Not Confirmed List Dialog Box);

    handle events:
        1. Display Matrix
        2. Print List
        3. Close

    get ( choice);
    switch ( choice ){
    case Display Matrix:
        call process Display matrix;
    case Print List:
        call process Print List;
    case  Close:
        call process Close;
    }
}
```

Process 2.20.2: Display Matrix

```
{
    display ( selected student's Student Info in Student Info Dialog Box);
}
```

Process 2.20.3: Print List

```
{
    print (  students currently displayed );
}
```

Process 2.20.4: Close

```
{
    close  Not Confirmed List Dialog Box;
}
```

Process 2.22.1: Handle MailBox Events

```
{
    display ( Mailbox in SendMail Dialog Box);

    handle events:
        1. Send Mail
        2. Display Send Log
        3. Display Receive Log
        4. Close
        5. Help

    get ( choice);
    switch ( choice ){
    case Send Mail:
        call process Send Mail;
    case Display Send Log:
        call process Display Send Log;
    case  Display Receive Log:
        call process Display Receive Log;
    case  Close:
        call process Close;
    case  Help
        call process Help;
    }
}
```

Process 2.22.2: Send Mail

```
{
    if ( "Send to Individual" is chosen){
        get ( Student SSN);
        if  ( StudentSSN is valid){
            get ( message);
            send ( message ) to StudentSSN's mailbox;
        }
    }
    else{
        get ( message);
        send ( message ) to all students' mailboxes;
    }
}
```

Process 2.22.3: Display Send Log

```
{
    display (  Send Log in Send Log Dialog Box);

    handle events:
        1. Find Student
        2. Display Message
        3. Delete
        4. Close
        5. Help
    get ( choice);
    switch ( choice ){
    case Find Student:
        display ( student to whom the selected message is sent);
    case Display Message:
        display ( selected sent message) ;
    case  Delete:
        remove ( selected sent message from the Send Log);
    case  Close:
        close Send Log Dialog Box;
    case  Help
        display (Help Topic for Send Log);
    }
}
```

Process 2.22.4: Display Receive Log

```
{
    display ( Receive Log in Receive Log Dialog Box);
    handle events:
        1. Reply to the Message
        2. Find Student
        3. Display Message
        4. Delete
        5. Close
        6. Help
    get ( choice);
    switch ( choice ){
    case Reply to the Message
        open ( open "Send Message Dialog Box");
    case Find Student:
        display ( student from whom the selected message is received);
    case Display Message:
        display ( selected received message) ;
    case  Delete:
        remove ( selected received message from the Receive Log);
    case  Close:
        close Receive Log Dialog Box;
    case  Help
        display  (Help Topic for Receive Log);
    }
}
```

Process 2.22.5: Close

```
{
    close  Send Mail Dialog Box;
}
```

Process 2.22.6: Help

```
{
    display  (Help Topic for Send Mail);
}
```

APPENDIX C. DATA DICTIONARY

accept_signal = * signal showing that the person who is logging in is accepted to the system *

add_course_req = * action of the curriculum staff's choosing option for adding course to the courses file*

add_password_req = * action of the curriculum staff's choosing option for adding password to the passwords file*

add_student_req = * action of the curriculum staff's choosing option for adding student to the students file*

add_track_req = * action of the curriculum staff's choosing option for adding track to the tracks file*

back_to_old_req = * action of the curriculum staff's or student's choosing option for displaying the original matrix which was in the database when the matrix window was first opened*

change_password_req = * action of the student's choosing the option for changing his or her password*

change_track_req = * action of the student's choosing the option for changing his or her track*

close_req = * action of the curriculum staff's or student's choosing the option for closing any displayed dialog box*

confirm_next_quarter_req= * action of the student's choosing the option for confirming his or her next quarter's schedule*

confirmation = * the two lists sent by the registrar to the instructors currently teaching a class and to the students enrolling in any of the classes being offered currently to confirm that registrars files are correct *

course = 2{A..Z | a..z}2 + 4{0..9}4
* code number of a course *

courses = **{ course }**

course_enrollment = **{ sectionno + student_name + { course } }**

current_matrix = * currently displayed matrix of a student *

delete_course_req = * action of the curriculum staff's choosing the option for deleting course from the courses file*

delete_message1_req = * action of the student's choosing the option for deleting message1 from the mailbox*

delete_message2_req = * action of the student's choosing the option for deleting message2 from the mailbox*

delete_password_req = * action of the curriculum staff's choosing the option for deleting password from the passwords file*

delete_student_req = * action of the curriculum staff's choosing the option for deleting student from the students file*

delete_track_req = * action of the curriculum staff's choosing the option for deleting track from the tracks file*

deny_next_quarter_req = * action of the student's choosing the option for denying his or her next quarter's schedule*

display_change_req` = * action of the curriculum staff's choosing the option for changing the display type of the enrollment list*

display_course_list_req = * action of the curriculum staff's or student's choosing the option for displaying the course list*

display_default_matrix_req = * action of the curriculum staff's or student's choosing the option for displaying the default matrix *

display_enrollment_list_req = * action of the curriculum staff's choosing the option for displaying the enrollment list *

display_mailbox_req = * action of the curriculum staff's or student's choosing the option for displaying opening the mailbox *

display_matrix_req = * action of the curriculum staff's or student's choosing the option for displaying the matrix*

display_more_info = * action of the curriculum staff's or student's choosing the option for displaying more information about an item in a list box*

display_not_ confirmed_list_req = * action of the curriculum staff's choosing the option for displaying the list of students who did not confirm their next quarters' schedule *

display_password_list = * action of the curriculum staff's choosin the option for displaying the password list*

display_personal_info_req= * action of the curriculum staff's or student's choosing the option for displaying the student information or personal information*

display_privileges_req = * action of the curriculum staff's or student's choosing the option for displaying the privileges *

display_receive_log_req = *action of the curriculum staff's choosing the option for displaying the receive log*

display_send_log_req = * action of the curriculum staff's choosing the option for displaying the send log*

display_settings = * action of the curriculum staff's choosing the option for displaying the settings*

display_track_list = * action of the curriculum staff's or student's choosing the option for displaying the track list*

enrollment_list = * a list showing the list of courses that the students are enrolling for together with the students' information *

final_offering = { **course** } + **sections** + **professors**
* issued by the department to the curriculum offices *

find_student_req = * action of the curriculum staff's choosing the option for finding the student that has the selected password in displayed password list dialog box*

generate_enrollment_list_req= * action of the curriculum staff's choosing the option for generating the enrollment list*

help_req = * action of the curriculum staff's or student's choosing the option for displaying the related help topic*

individual_matrix_changes= * changes made to a student's matrix by the student or by the curriculum staff*

matrix_info_and_changes= **general_info** + **individual_matrix_changes**

more_info_req = * action of the curriculum staff's or student's choosing the option for displaying more information about a selected item in a list box*

new_password = * password entered by the student when he or she wanted to change the previous password*

offered_courses = { course }

password = 1 { A..B | a..b | 0..9 } 8

passwords = { **password** }

prerequisite_flag = [true | false]

print_list_req = * action of the curriculum staff's choosing the option for printing the list of students or courses or enrollment list or students_not_confirmed_list or action of student's choosing the option for printing the list of courses*

print_matrix_req = * action of the curriculum staff's or student's choosing the option for printing the currently displayed matrix*

printout_of_course_list = * hard copy of the list of courses in the database*

printout_of_default_matrix =* hard copy of the default matrix currently displayed*

printout_of_enrollment_list =* hard copy of the enrollment list in the database*

printout_of_student_list = * hard copy of the list of the students in the curriculum*

printout_of_students_not_confirmed = * hard copy of the list of the students in the curriculum*

professors = {title + first_name + last_name}

quit_application_req = * action of the curriculum staff's or the student's choosing the option for quitting the application*

registrar_info = * any information coming from the registrar *

reject_signal = * signal showing that the person who is logging in is accepted to the system *

rooms = 1 { A..B } 2 + 3 { 1..9 } 3

save_req = * action of the curriculum staff's choosing the option for saving information about a student, a course, a track a default matrix, the settings or student privileges*

save_matrix_req = * action of the curriculum staff's or the student's choosing the option for saving the student's matrix*

schedule = * a list consisting of students, professors, courses rooms and time period *

schedule_info = * information consisting of enrollment list, instructors list and rooms list *

search_course_req = * action of the curriculum staff's choosing the option for searching a course from the database*

search_student_by_name_req = * action of the curriculum staff's choosing the option for searching a student by his or her first name or last name or both*

search_student_by_ssn_req = * action of the curriculum staff's choosing the option for
searching a student by his or her SSN*

sections = 1..3

select_track_req = * action of the student's choosing the option for selecting a track*

set_flags_to_false_req = * action of the curriculum staff's choosing the option for setting the confirmation flags for all students to false*

settings_info = * any kind of information written to or retrieved from by curriculum staff*

send_mail_req = *action of the curriculum staff's choosing the option for sending mail to the student(s) *

student_info = * any information about the student except his matrix *

student_academic_info = * any academic information about a student *

system_info = * any type of information that the student is getting while he is working on his matrix *

system_updates = * any type of updates made to the files in the system by curriculum staff*

title = [Prof | Doc | Associate Prof]

track = 1{ A..Z | a..z } 50 + 1{1..8 + 1{ **course** }6 }8 + **track_electives**

tracks = { **track** }

track_electives = 1 { course }15

APPENDIX D. USER MANUAL

A. CROSS REFERENCE

B. INSTALLATION GUIDE

The ACES application can be installed to the computer by running the "install.bat" and "setdll.bat" files in installation diskettes.

1. Installation for Server or a Stand-alone Computer

This type of installation creates the "C:\aces" directory and copies the necessary files to this directory.

a. Installing from Windows File Manager

- Put the Installation Diskette 1 to "A" or "B" drive of your computer.
- Display the "A" or "B" directory depending on the floppy disk drive that you install from in the File Manager window.
- Locate the the "install.bat" file.
- Double-click the "install.bat" file or choose "Run" from the "File" menu after highlighting the "install.bat" file.
- To install the "DLL" files, use the Installation Diskette 2.
- Follow the same procudure as described above, but this time run the "setdll.bat" file.

For a moment the "MS-DOS window will appear, and later disappear after application is installed.

b. Installing from Windows Program Manager

- Put the Installation Diskette 1 to "A" or "B" drive of your computer.
- Choose "Run" command from the "File" menu command in Program Manager.
- Write "install.bat" in the space provided and press enter.
- To install the "DLL" files, use the Installation Diskette 2.
- Follow the same procudure as described above, but this time run the "setdll.bat" file.

For a moment the "MS-DOS window will appear, and later disappear after application is installed.

c. Installing from the Ms-Dos prompt

- Put the Installation Diskette 1 to "A" or "B" drive of your computer.

- Write "install.bat" in the command line and then press enter.
- To install the "DLL" files, use the Installation Diskette 2.
- Follow the same procudure as described above, but this time run the "setdll.bat" file.

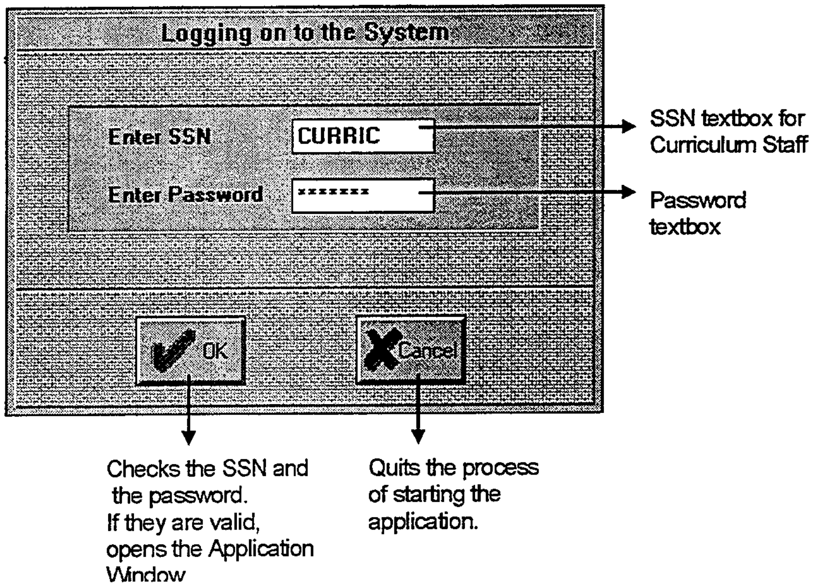
2. Installation For Client Computers

This type of installation also creates the "C:\aces" directory and copies the necessary files to this directory. But in this case, some files that are copied to the server computer are not copied to this computer. Rest of the installation process is the same procedure explained in the previous section.

C. USER MANUAL FOR THE CURRICULUM STAFF

1. Logging On To The System

The curriculum staff starts the application by double clicking the application icon like any other applications in Windows Program Manager. The first dialog box which is shown in Figure D1 pops up in the center of the screen, waiting for the user to enter the "SSN" and the "Password". The "SSN" for the curriculum staff is hard-coded as "CURRIC" into the program and can not be changed without recompiling the program. This is how the program understands that somebody is trying to access to the system with the privileges given only to the curriculum staff. But the password is not constant and can be changed by the curriculum staff once they are in the program. A temporary password has been designated by the programmer of the application to be able to start the application for the first time.



Logging on to the System

Enter SSN CURRIC

Enter Password *******

SSN textbox for Curriculum Staff

Password textbox

Checks the SSN and the password. If they are valid, opens the Application Window

Quits the process of starting the application.

Figure D1. Log On Dialog Box

2. Application Window

When the "OK" button is pressed in "Log on Dialog Box", the application window shown in Figure D2 is displayed, if the password is correctly entered. In this application window, there is a menu that includes all of the functions that the curriculum staff needs to be able to interact with the system. A speedbar is also displayed right under the menu bar in order to provide easy access to the same functions in the menu. In the status bar, the description of either the menu commands or the speed bar commands are displayed dynamically as the mouse pointer moves over the speed bar or the menu commands.

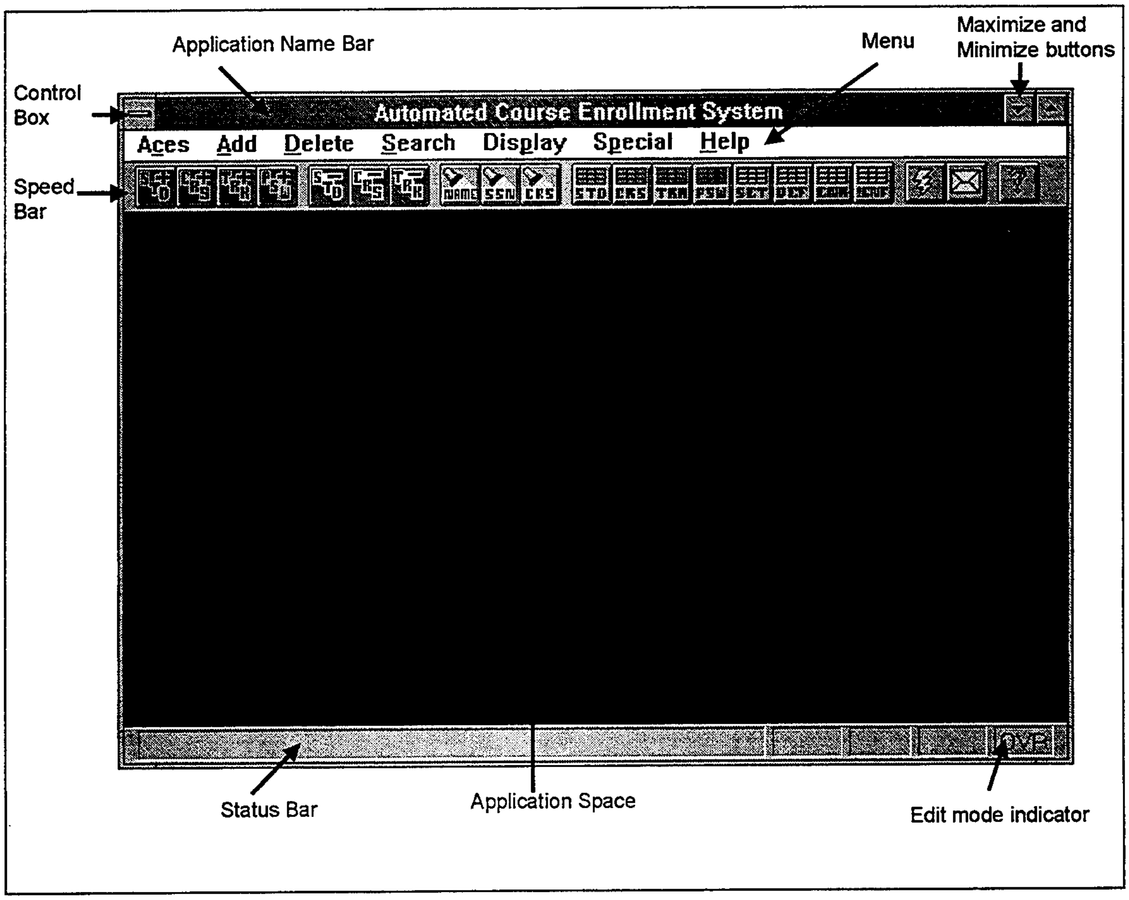


Figure D2. Application Window for Curriculum Staff

3. Menu Commands

In Figure D3, the menu commands are displayed.

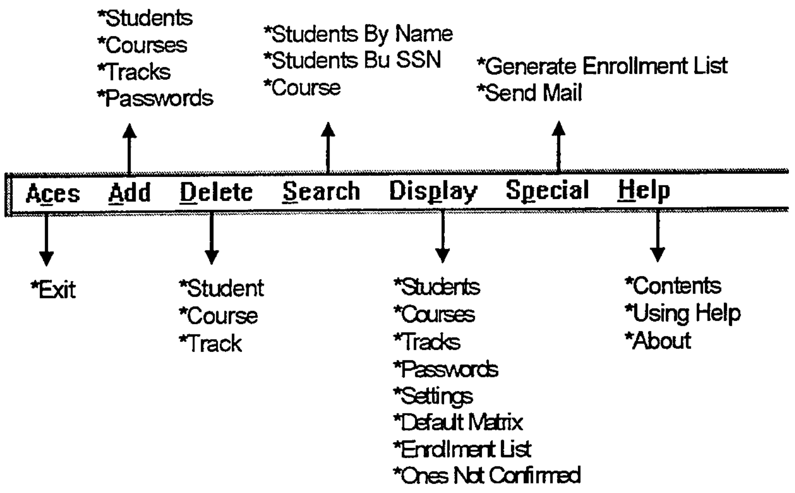


Figure D3. Menu Commands

4. Speed Bar Commands

In Figure D4, Speed Bar commands are shown.

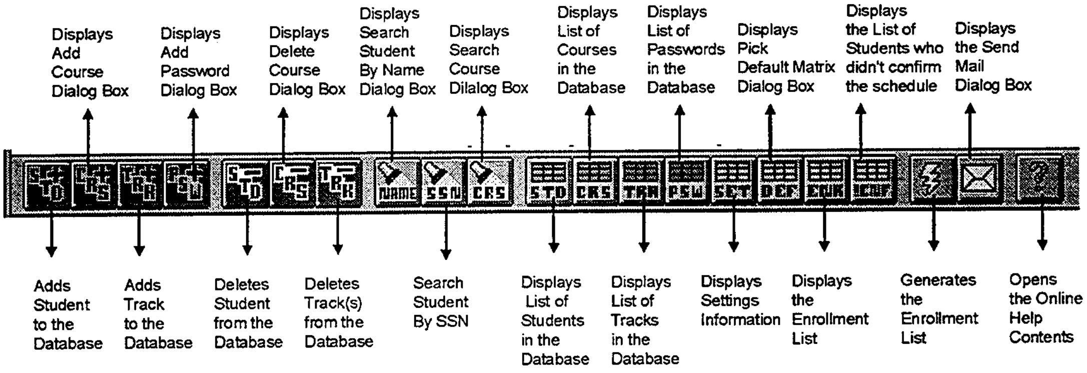

Displays Add Course Dialog Box
Displays Add Password Dialog Box
Displays Delete Course Dialog Box
Displays Search Student By Name Dialog Box
Displays Search Course Dilalog Box
Displays List of Courses in the Database
Displays List of Passwords in the Database
Displays Pick Default Matrix Dialog Box
Displays the List of Students who didn't confirm the schedule
Displays the Send Mail Dialog Box
NAME
SSN
CRS
STD
CRS
TRK
PSW
SET
DEF
ENR
ICNF
Adds Student to the Database
Adds Track to the Database
Deletes Student from the Database
Deletes Track(s) from the Database
Search Student By SSN
Displays List of Students in the Database
Displays List of Tracks in the Database
Displays Settings Information
Displays the Enrollment List
Generates the Enrollment List
Opens the Online Help Contents

Figure D4. Speed Bar Commands

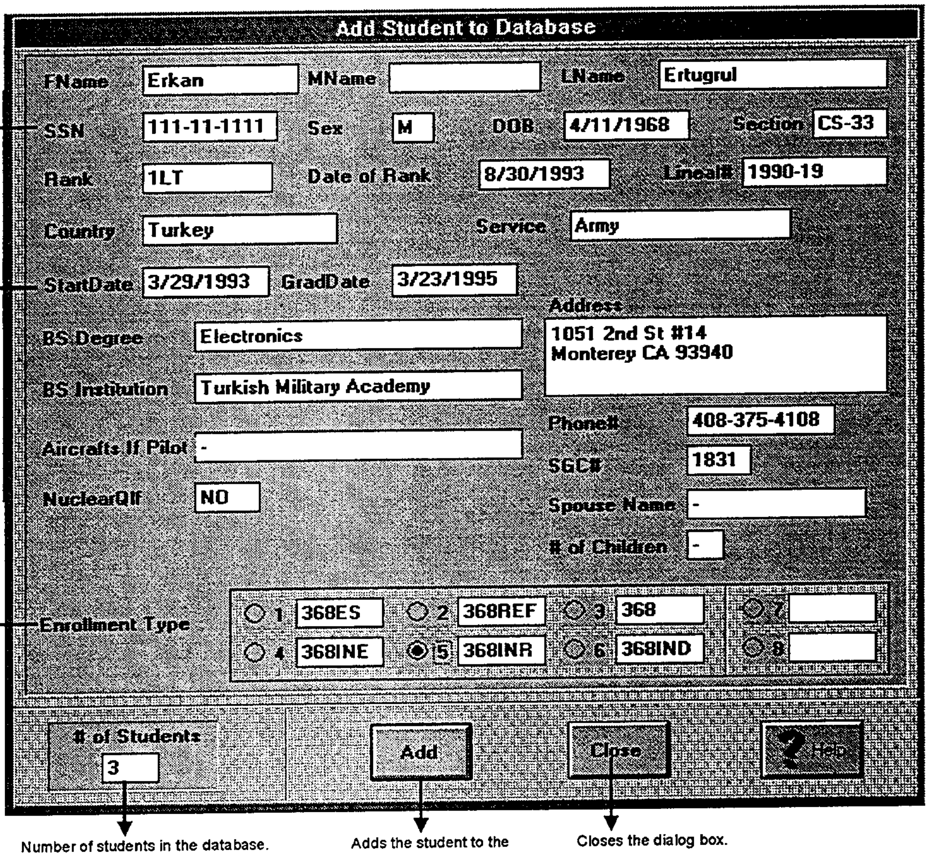
5. Add Student Dialog Box

If "Add Student" command is selected from either the menu or the speed bar, the "Add Student Dialog Box" is displayed as shown in Figure D5. There is a number of edit boxes for entering information about the student. But, not all of those edit boxes have to be entered to in order to add a new student to the database. Only "SSN" and "Enrollment Type" have to be entered, the other information is optional. They can be entered later by displaying the "Student Information Dialog Box". Pressing "Add" button would add a new student to the database with the information entered in the edit boxes, if the entries are valid. Since "SSN" is the key field of the "Students File", no duplication would be allowed for this field. The SSN also has to be in the correct format. The format is like in "111-11-1111". The date format also has to be in "mm/dd/yy" or "mm/dd/yyyy" format. Pressing the "Close" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Add Student Dialog Box".

SSN has to be entered since it is key field of the students database.

This is the date of the first quarter of the student, not the start date of his education in the school. This date is very important, because it is used to calculate the current quarter of the student at any time. Format is like: MM/DD/YYYY

This shows the type of the defaut matrix that the student will be assigned. The names for each kind of default matrix is decided by the curriculum officer and edited and stored in Settings Dialog Box.



FName: Erkan MName: LName: Ertugrul

SSN: 111-11-1111 Sex: M DOB: 4/11/1968 Section: CS-33

Rank: 1LT Date of Rank: 8/30/1993 Lineal#: 1990-19

Country: Turkey Service: Army

StartDate: 3/29/1993 GradDate: 3/23/1995

Address:
1051 2nd St #14
Monterey CA 93940

BS Degree: Electronics

BS Institution: Turkish Military Academy

Phone#: 408-375-4108

Aircrafts If Pilot: -

SGC#: 1831

NuclearQlf: NO

Spouse Name: -

# of Children: -

Enrollment Type:
○ 1 368ES ○ 2 368REF ○ 3 368 ○ 7
○ 4 368INE ● 5 368INR ○ 6 368IND ○ 8

# of Students: 3

Number of students in the database. It also serves as a feedback, when the students are entered to the database.

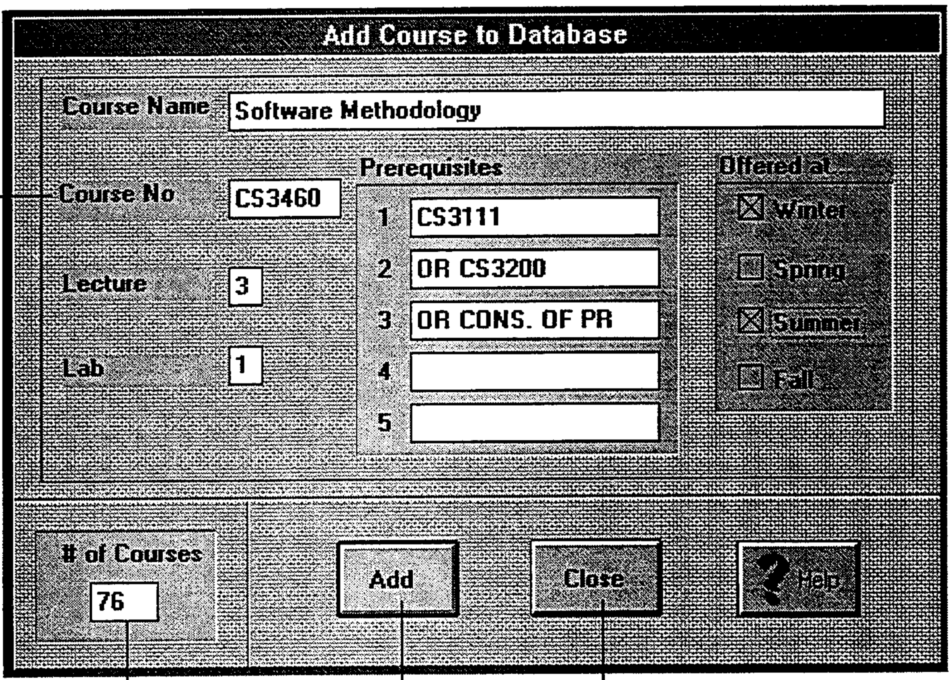
Adds the student to the database with the information currently displayed.

Closes the dialog box.

Figure D5. Add Student Dialog Box

6. Add Course Dialog Box

If "Add Course" command is selected from either the menu or the speed bar, the "Add Course Dialog Box" is displayed as shown in Figure D6. There is a number of edit boxes each of which are supposed to get some information about the course. But, not all of those edit boxes have to be entered to be able to add a new course to the database. Only "Course No" has to be entered, the other information is optional. They can be entered later by displaying the "Course Information Dialog Box". Pressing "Add" button would add a new course to the database with the information entered in the edit boxes, if the entries are valid. Since "Course No" is the key field of the "Courses File", no duplication would be allowed for this field. Pressing the "Close" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Add Course Dialog Box".



Add Course to Database

Course Name: Software Methodology

Course No: CS3460

Lecture: 3

Lab: 1

Prerequisites
1 CS3111
2 OR CS3200
3 OR CONS. OF PR
4
5

Offered at
☒ Winter
□ Spring
☒ Summer
□ Fall

# of Courses: 76

Course No has to be entered since it is the key field of Courses File

This shows the number of courses in the database dynamically. It is also a feedback to show whether the courses are being added or not.

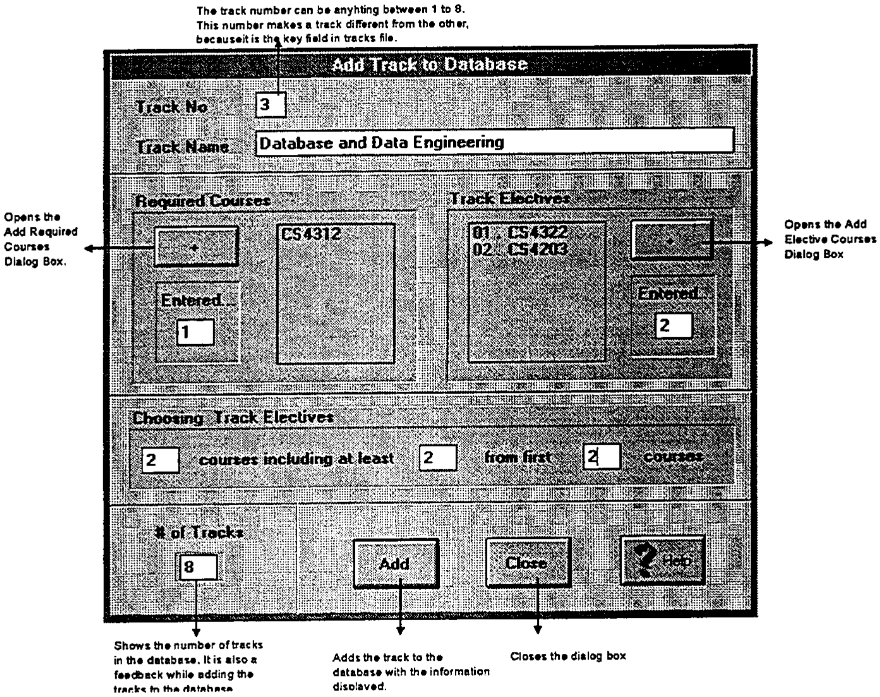
Adds the course to the database with the information currently displayed.

Closes the dialog box.

Figure D6. Add Course Dialog Box

7. Add Track Dialog Box

If "Add Track" command is selected from either the menu or the speed bar, the "Add Track Dialog Box" is displayed as shown in Figure D7. There is a number of edit boxes each of which are supposed to get some information about the track. But, not all of those edit boxes have to be entered to be able to add a new track to the database. Only "Track No" has to be entered, the other information is optional. They can be entered later by displaying the "Track Information Dialog Box". Pressing "Add" button would add a new track to the database with the information entered in the edit boxes, if the entries are valid. Since "Track No" is the key field of the "Courses File", no duplication would be allowed for this field. The edit boxes in the section of the dialog box determine the rules regarding the selection of "Track Electives" from the "Track Electives List". The "Required Courses" and the "Track Electives" are entered by pressing the "+" Button in the corresponding sections. The dialog boxes displayed to enter required and elective courses are displayed in Figure D28 and Figure D29 respectively.


The track number can be anyhting between 1 to 8. This number makes a track different from the other, becauseit is the key field in tracks file.

Add Track to Database

Track No 3

Track Name Database and Data Engineering

Required Courses: CS4312

Entered 1

Track Electives: 01 CS4322
02 CS4203

Entered 2

Choosing Track Electives

2 courses including at least 2 from first 2 courses

# of Tracks 8

Opens the Add Required Courses Dialog Box.

Opens the Add Elective Courses Dialog Box

Shows the number of tracks in the database. It is also a feedback while adding the tracks to the database

Adds the track to the database with the information displayed.

Closes the dialog box

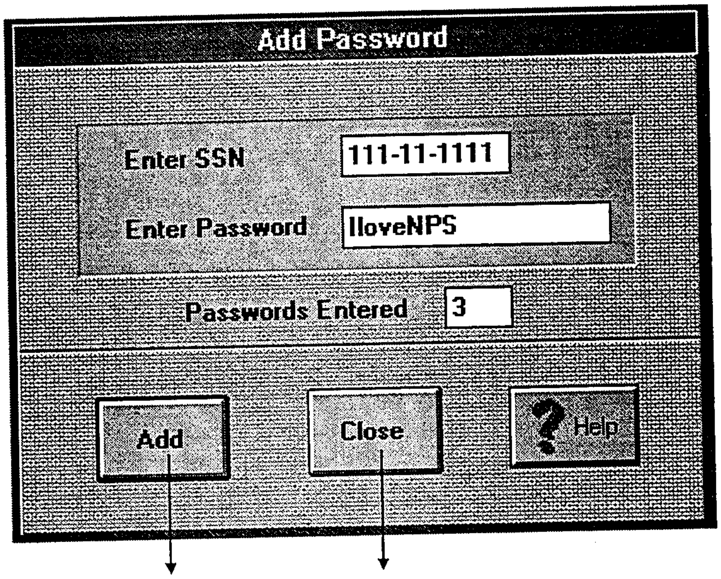
Figure D7. Add Track Dialog Box

8. Add Password Dialog Box

If "Add Password" command is selected from either the menu or the speed bar, the "Add Password Dialog Box" is displayed as shown in Figure D8. There are two edit boxes. One is "SSN" and the other is "Password". Pressing "Add" button would add a new password to the database with the information entered in the edit boxes, if the entries are valid. Since "Password No" is the key field of the "Passwords File", no duplication would be allowed for this field. Under the following conditions the process of adding a password to the database is not completed successfully.

- If the entered "SSN" is not in the database.
- If the entered "SSN" already has a password.
- If the "Password" is already assigned to somebody else.

Pressing the "Close" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Add Password Dialog Box".



Enter SSN 111-11-1111

Enter Password IloveNPS

Passwords Entered 3

Adds the password to the database with the information displkayed

Closes the dialog box

Figure D8. Add Password Dialog Box

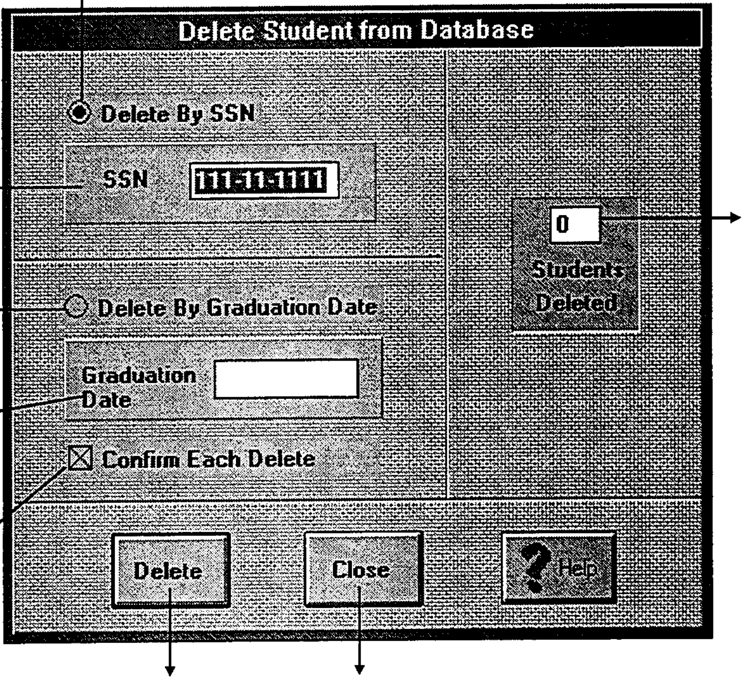
9. Delete Student Dialog Box

If "Delete Student" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D9 is displayed. Two options are given as to deleting the students from the database:

- Deleting one student by entering his "SSN" and then pressing the "Delete" button.
- Deleting a group of students, by entering their graduation date and then pressing the "Delete" button. This option is useful to delete the graduating students all together.

It is important to note that, deleting a student from the database will also delete the corresponding passwords from the database automatically. If the "Delete By Graduation" option is selected, then the curriculum staff has the chance to confirm each delete to make sure that they are not making wrong delete by checking the "Confirm Each Delete" check box. By default, this check box is checked. Pressing the "Close" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Delete Student Dialog Box".

This option is checked if the curriculum staff wants to remove a student from the database by entering the student's SSN



Delete Student from Database

● Delete By SSN

SSN 111-11-1111

○ Delete By Graduation Date

Graduation Date

☒ Confirm Each Delete

0 Students Deleted

SSN has to be entered if "Delete by SSN" option is selected.

This option is checked if the curriculum staff wants to remove a group of students from the database by entering the graduation date

Graduation Date has to be entered if "Delete by Graduation Date" option is selected.

This checkbox is checked if it is desired to delete a group of students by confirming each of the deletes.

Displays the number of students deleted in one session

Deletes a student whose SSN is entered if "Delete By SSN option is checked. Or deletes a group of students whose graduation dates are entered if "Delete By Graduation Date" option is checked.

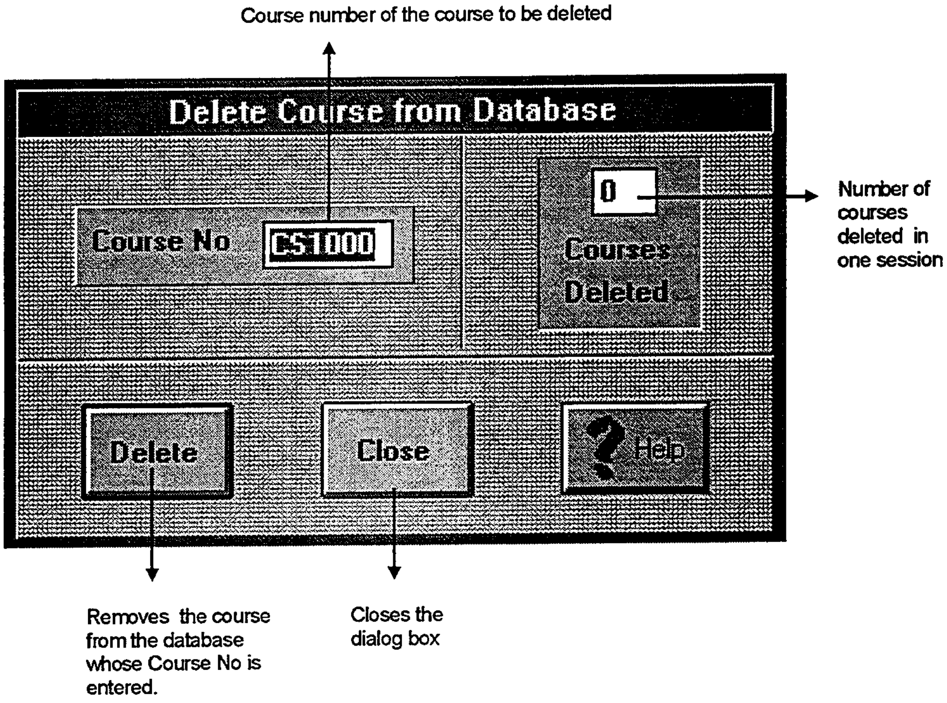
Closes the dialog box

Figure D9. Delete Student Dialog Box

10. Delete Course Dialog Box

If "Delete Course" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D10 is displayed. In order to be able to remove a course from the database, the entered "Course No" has to be in the correct form and be in the database. When the "Delete" button is pressed a confirmation question regarding the delete operation is asked before each delete. It is important to note that, a course is not allowed to be deleted if it is a track requirement or a track elective for a track or a course in one of the default matrices. The user is first prompted to remove those courses from the tracks or default matrices found.

Pressing the "Close" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Delete Course Dialog Box".



Delete Course from Database

Course No CS1000

0 Courses Deleted

Course number of the course to be deleted

Number of courses deleted in one session

Removes the course from the database whose Course No is entered.

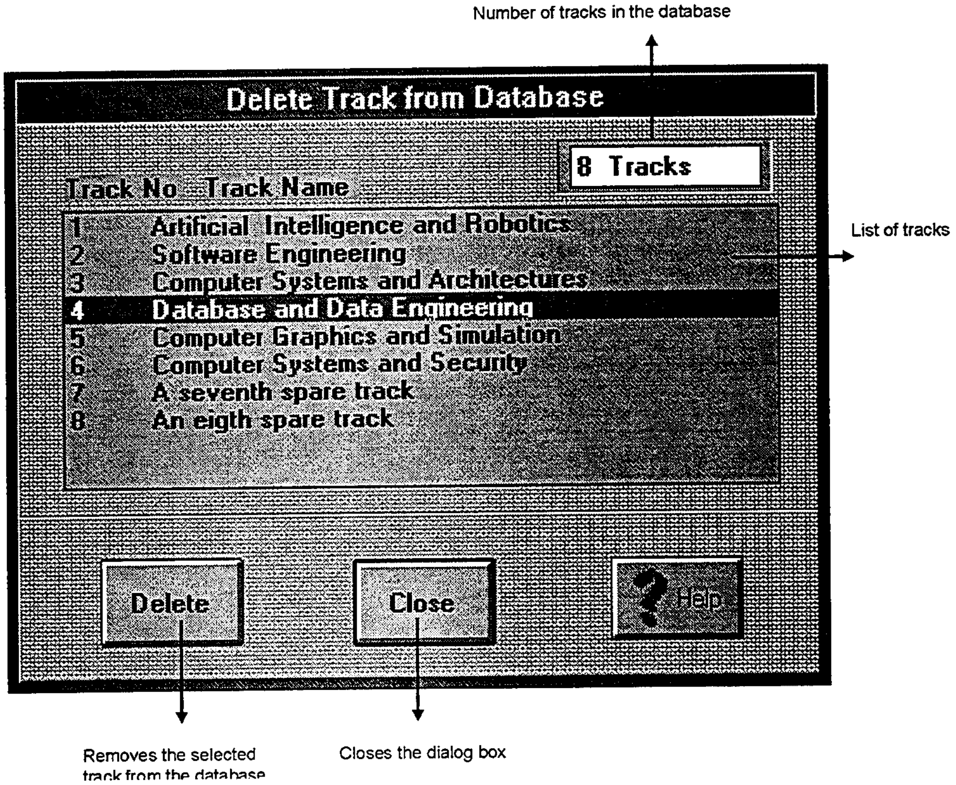
Closes the dialog box

Figure D10. Delete Course Dialog Box

11. Delete Track Dialog Box

If "Delete Track" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D11 is displayed. The track to be deleted has highlighted in the list by clicking on the track name. When the "Delete" button is pressed a confirmation question regarding the delete operation is asked to the user before the actual delete takes place.

Pressing the "Close" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Delete Track Dialog Box".



Number of tracks in the database

Delete Track from Database

8 Tracks

Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track
8	An eigth spare track

List of tracks

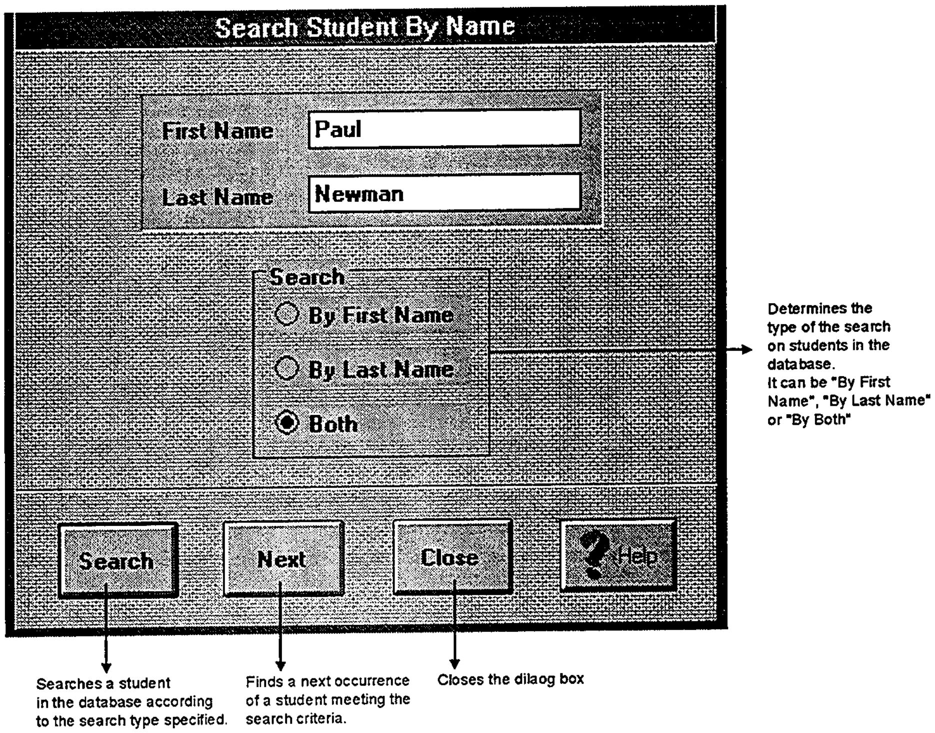
Removes the selected track from the database

Closes the dialog box

Figure D11. Delete Track Dialog Box

12. Search Student By Name Dialog Box

If "Search Student By Name" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D12 is displayed. Three options are provided when searching for a student in the database. A student can be searched by the "First Name" or the "Last Name" or the "First Name and Last Name" together. Choosing the last option reduces the scope of the search, but increases the chance of misspelling a name. If the user knows the first and the last name of the student for sure, then it is the fastest way to go to a student's record. If one of the first two options are being used, then using the "Next" functionality may prove to be effective, since there may be more than one student with the same first name or last name. Pressing the "Search" button starts the search according to the given criteria and pressing the "Next" button makes the search for the next instance of the given criteria in the database. When the "Close" button is pressed, the dialog box is closed and when the "Help" button is pressed, the on-line help for the "Search Student By Name Dialog Box" is displayed.



First Name: Paul

Last Name: Newman

Search
○ By First Name
○ By Last Name
● Both

Determines the type of the search on students in the database. It can be "By First Name", "By Last Name" or "By Both"

Searches a student in the database according to the search type specified.

Finds a next occurrence of a student meeting the search criteria.

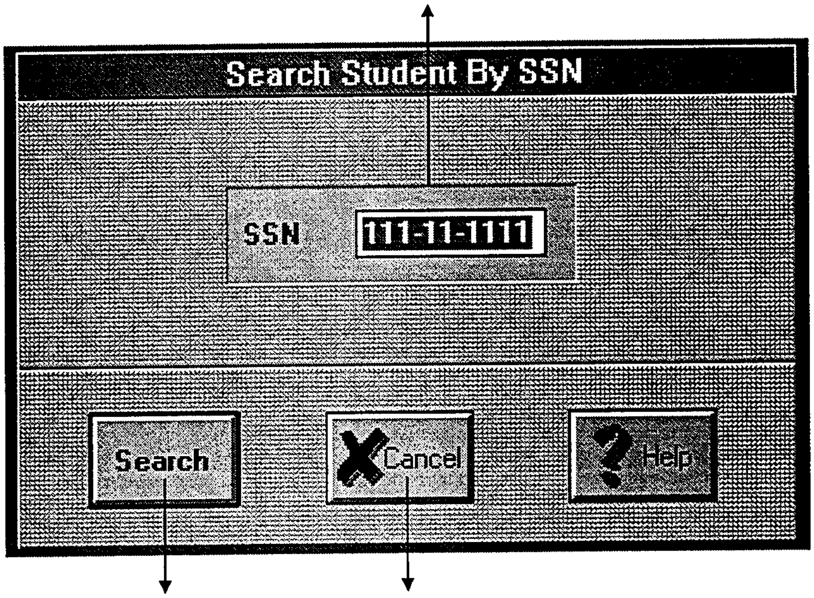
Closes the dilaog box

Figure D12. Search Student By Name Dialog Box

13. Search Student By Ssn Dialog Box

If "Search Student By SSN" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D13 is displayed. In order to be able to search a student correctly, the SSN entered has to be valid and belong to a student in the database. Since the SSN is the key field in the Students File, it is impossible to have more than one student having the same SSN. If the student is found in the database, then the student information is displayed in "Student Info Dialog Box" shown in Figure D16. Pressing the "Search" button starts the search according to the given criteria. When the "Close" button is pressed, the dialog box is closed and when the "Help" button is pressed, the on-line help for the "Search Student By SSN Dialog Box" is displayed.

SSN of the student to be searched



SSN 111-11-1111

Searches the student in the database whose SSN is entered.

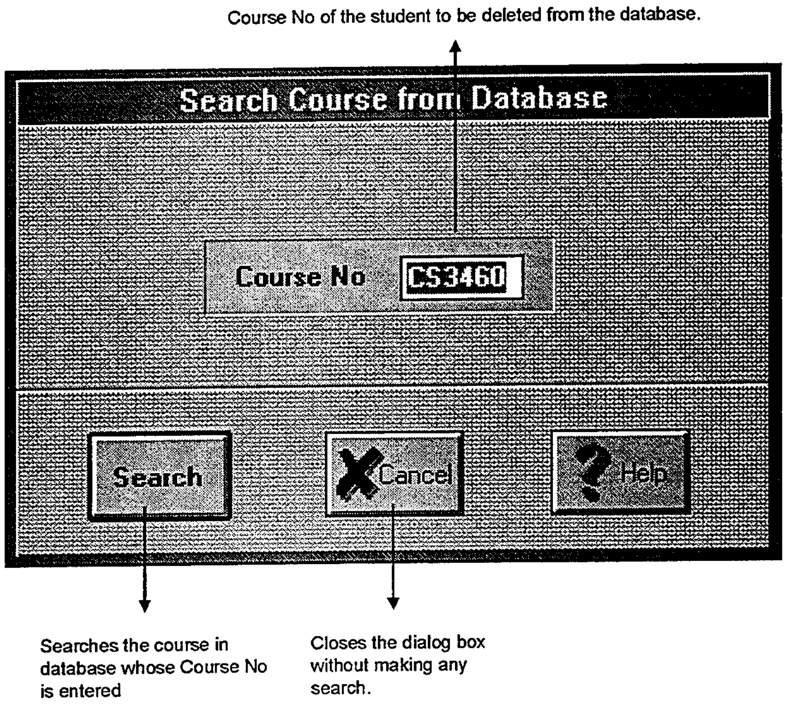
Closes the dialog box without making any search.

Figure D13. Search Student By SSN

14. Search Course Dialog Box

If "Search Course" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D14 is displayed. In order to be able to search a course successfully, the entered "Course No" has to be valid. If the course is found in the database, the course information is displayed in "Course Information Dialog Box" shown in Figure D25.

Pressing the "Cancel" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Search Course Dialog Box".



Course No of the student to be deleted from the database.

Course No CS3460

Searches the course in database whose Course No is entered

Closes the dialog box without making any search.

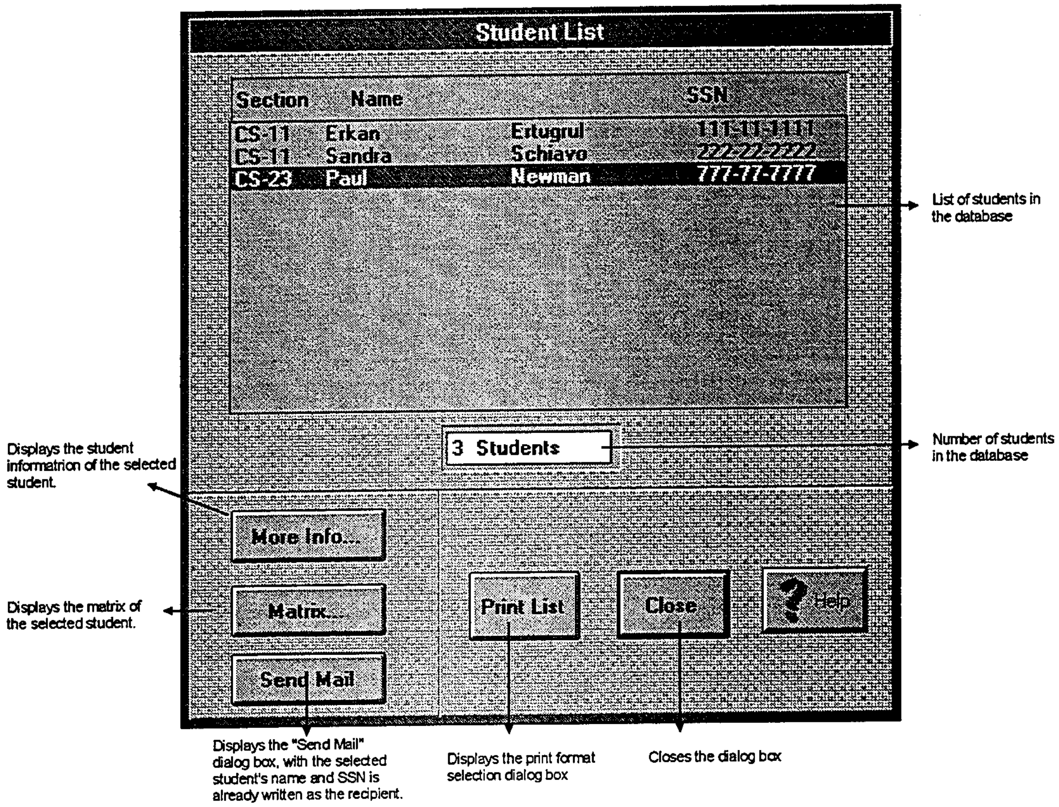
Figure D14. Search Course Dialog Box

15. Student List Dialog Box

If "Display Students" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D15 is displayed. Once the list of the students is displayed, the user can do the following things:

- Getting more information about the student highlighted in the list and displaying it in the "Student Info Dialog Box" shown in Figure D16 by pressing the "More Info..." button.
- Getting the matrix of the student highlighted in the list and displaying it in the "Student Matrix Dialog Box" shown in figure D17 by pressing the "Matrix..." button.
- Opening the "Send Mail Dialog Box" shown in Figure D36 by pressing the "Send Mail" button to be able to send mail to the student highlighted in the list.
- Displaying the options of print formats and then printing the displayed student list by pressing the "Print List" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Student List Dialog Box" by pressing "Help" button.

Double-clicking a student's entry in the list also gets more information about this student.



Student List

Section	Name		SSN
CS-11	Erkan	Ertugrul	111-11-1111
CS-11	Sandra	Schiavo	222-22-2222
CS-23	Paul	Newman	777-77-7777

3 Students

List of students in the database

Number of students in the database

Displays the student informatrion of the selected student.

Displays the matrix of the selected student.

Displays the "Send Mail" dialog box, with the selected student's name and SSN is already written as the recipient.

Displays the print format selection dialog box

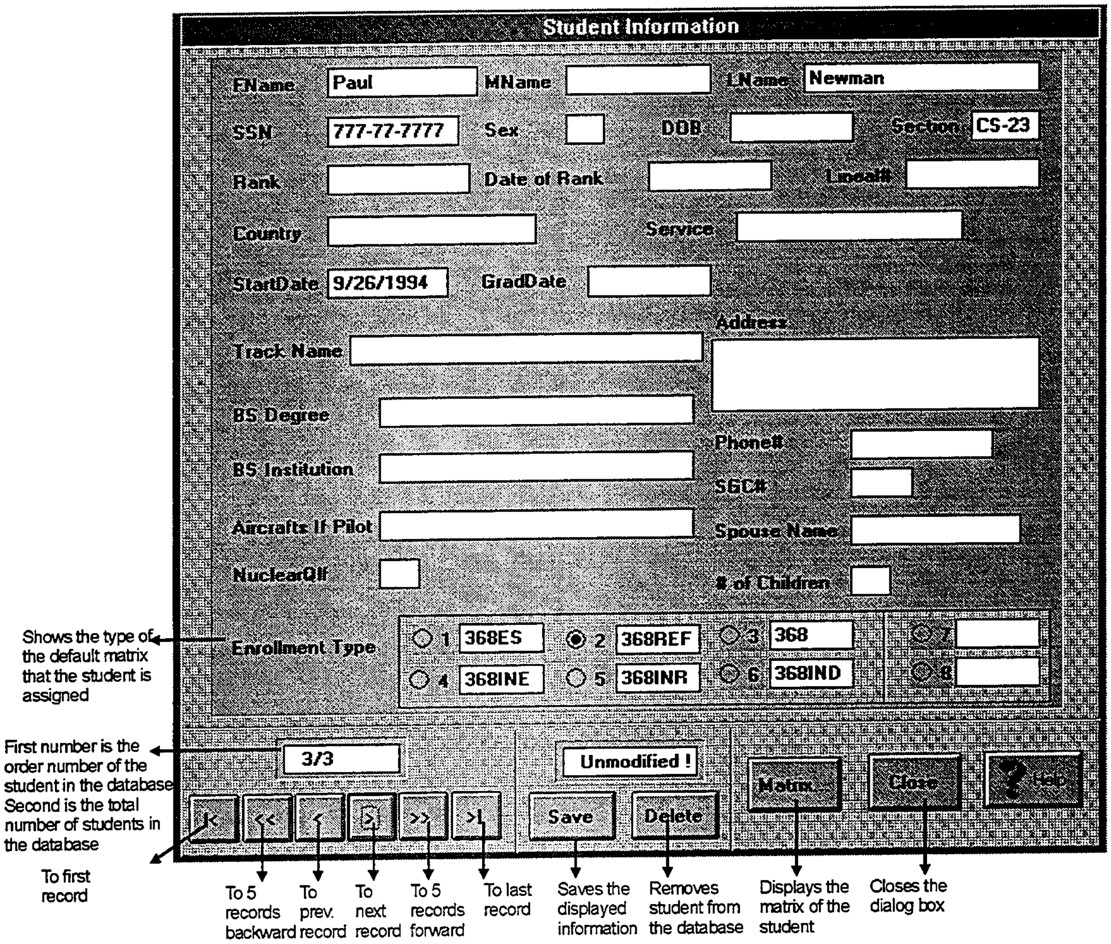
Closes the dialog box

Figure D15. Student List Dialog Box

16. Student Information Dialog Box

If "More Info..." command is selected from the "Student List Dialog Box" or from some other dialog box, the dialog box shown in Figure D16 is displayed. Once the information of a student is displayed, the user can do the following things:

- Modifying the student's information except the "Track Name".
- Going to other student records by using the "Navigation Buttons".
- Deleting the displayed student from the database by pressing the "Delete" button. Deleting a student will also cause the student's password be deleted.
- Saving the changes made by pressing the "Save" button.
- Displaying the matrix of the displayed student in "Student Matrix Dialog Box" shown in Figure D17 by pressing the "Matrix" button.
- Closing the dialog box and displaying the on-line help.



FName Paul
MName
LName Newman
SSN 777-77-7777
Sex
DOB
Section CS-23
Rank
Date of Rank
Lineal#
Country
Service
StartDate 9/26/1994
GradDate
Address
Track Name
BS Degree
BS Institution
Aircrafts If Pilot
NuclearQlf
Phone#
SGC#
Spouse Name
# of Children
Enrollment Type ○ 1 368ES ● 2 368REF ○ 3 368 ○ 7
○ 4 368INE ○ 5 368INR ○ 6 368IND ○ 8

3/3

Unmodified !

Shows the type of the default matrix that the student is assigned

First number is the order number of the student in the database Second is the total number of students in the database

To first record

To 5 records backward

To prev. record

To next record

To 5 records forward

To last record

Saves the displayed information

Removes student from the database

Displays the matrix of the student

Closes the dialog box

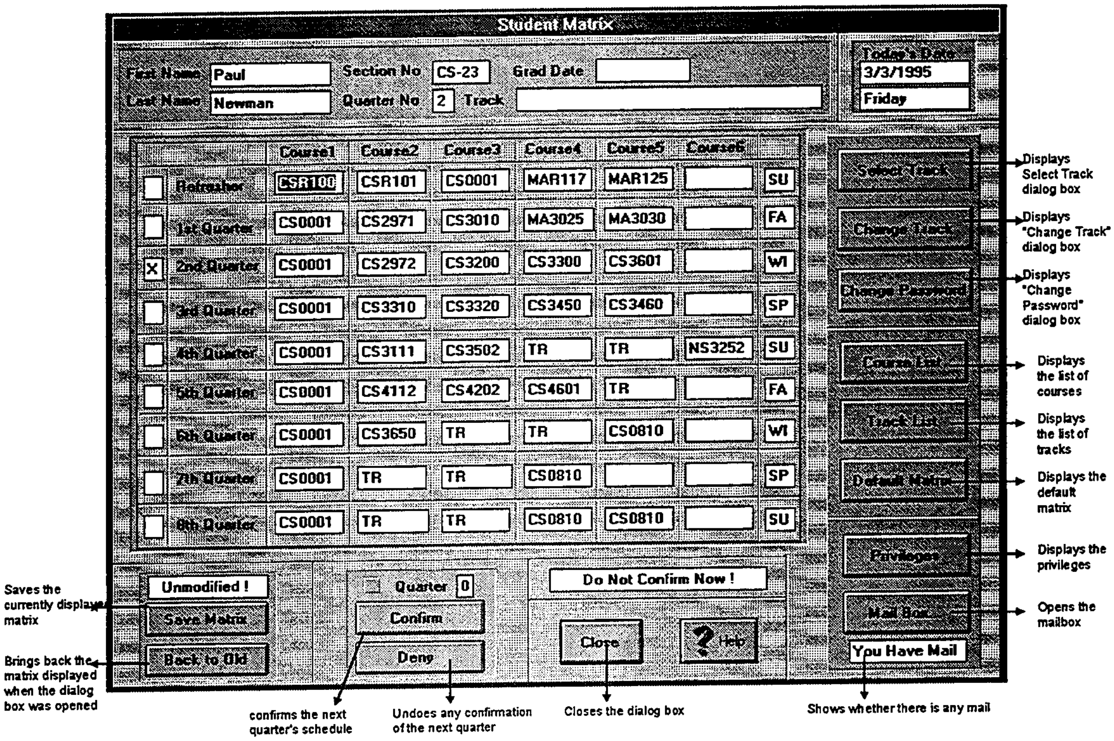
Figure D16. Student Information Dialog Box

17. Student Matrix Dialog Box

If "Matrix..." command is selected from the "Student List Dialog Box" or from some other dialog box, the dialog box shown in Figure D17 is displayed. Once the matrix of a student is displayed, the user can do the following things:

- Modifying the student's matrix.
- Selecting a track, if no track is selected yet, by pressing the "Select Track" button and then displaying the "Select Track Dialog Box" shown in Figure D18.
- Changing the track selected before, by pressing the "Change Track" button and then displaying the "Change Track Dialog Box" shown in Figure D19.
- Changing the password of the student, by pressing the "Change Password" button and then displaying the "Change Password Dialog Box" shown in Figure D20.
- Displaying the course list in the "Course List Dialog Box" shown in Figure D24, by pressing the "Course List" button..
- Displaying the track list in the "Track List Dialog Box" shown in Figure D26, by pressing the "Track List" button.
- Displaying the default matrix of the student in the "Default Matrix Dialog Box" shown in Figure D33, by pressing the "Default Matrix" button.
- Displaying the privileges of the student in "Privileges Dialog Box" shown in Figure D21, by pressing the "Privileges" dialog box.
- Displaying the mailbox of the student in "Student Mail Box" shown in Figure D22, by pressing the "Mail Box" button.
- Confirming the next quarter's schedule for the displayed student by pressing the "Confirm" button.
- Denying the next quarter's schedule for the displayed student by pressing the "Deny" button.
- Saving the matrix by pressing the "Save" button.
- Bringing back the matrix that was first displayed when the "Student Matrix Dialog Box" was opened.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Student Matrix Dialog Box" by pressing the "Help" button.

When modifying the matrix of the student, no restriction is applied against the curriculum staff. They can modify any part of the matrix, but they can not enter a course in a matrix that does not exist in the database. Although the curriculum staff can do any changes in the matrix, including the selection and change of the track, it is not recommended to make changes in the matrix without the knowledge of the student. Otherwise students may get confused with their own matrices. For that reason, this dialog box should basically used for viewing purposes.



Student Matrix

First Name: Paul
Last Name: Newman
Section No: CS-23
Quarter No: 2
Grad Date:
Track:
Today's Date: 3/3/1995
Friday

		Course1	Course2	Course3	Course4	Course5	Course6	
☐	Refresher	CSR100	CSR101	CS0001	MAR117	MAR125		SU
☐	1st Quarter	CS0001	CS2971	CS3010	MA3025	MA3030		FA
☒	2nd Quarter	CS0001	CS2972	CS3200	CS3300	CS3601		WI
☐	3rd Quarter	CS0001	CS3310	CS3320	CS3450	CS3460		SP
☐	4th Quarter	CS0001	CS3111	CS3502	TR	TR	NS3252	SU
☐	5th Quarter	CS0001	CS4112	CS4202	CS4601	TR		FA
☐	6th Quarter	CS0001	CS3650	TR	TR	CS0810		WI
☐	7th Quarter	CS0001	TR	TR	CS0810			SP
☐	8th Quarter	CS0001	TR	TR	CS0810	CS0810		SU

Unmodified !
☐ Quarter 0
Do Not Confirm Now !
You Have Mail

Displays Select Track dialog box

Displays "Change Track" dialog box

Displays "Change Password" dialog box

Displays the list of courses

Displays the list of tracks

Displays the default matrix

Displays the privileges

Opens the mailbox

Saves the currently displayed matrix

Brings back the matrix displayed when the dialog box was opened

confirms the next quarter's schedule

Undoes any confirmation of the next quarter

Closes the dialog box

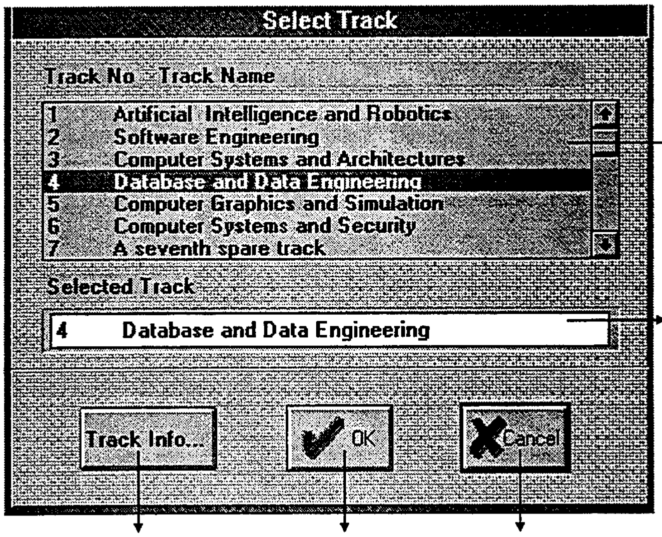
Shows whether there is any mail

Figure D17. Student Matrix Dialog Box

18. Select Track Dialog Box

If "Select Track" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D18 is displayed. In this dialog box, the user can do the following things:

- Getting more information about a track displayed in the list by first highlighting it and then pressing the "Track Info..." button.. The highlighted track name will be displayed in another text box just for a better feedback and if the "Track Info..." button is pressed, the information about the selected track will be displayed in "Track Info Dialog Box" shown in Figure D27.
- Selecting a track as the track of the student by first highlighting it and then pressing the "OK" button. There will be a confirmation question asked to the user. When it is confirmed, the track requirement courses in the selected track will be automatically copied to the student's matrix. The elective courses of the track will still have to be entered manually. The curriculum staff will not be asked to enter all of the track requirements while they are saving the new matrix, unlike the students who will not be allowed to save the changes made in the matrix before meeting all of the requirements of the selected track. For that reason, not to cause any confusion, the curriculum staff's selecting a track is not recommended.
- Canceling selecting a track and closing dialog box by pressing "Cancel" button.


Select Track

Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track

Selected Track

4 Database and Data Engineering

The list of valid tracks in the database

The selected track

Displays the track information for the selected track

The student's track becomes saved as the selected track here

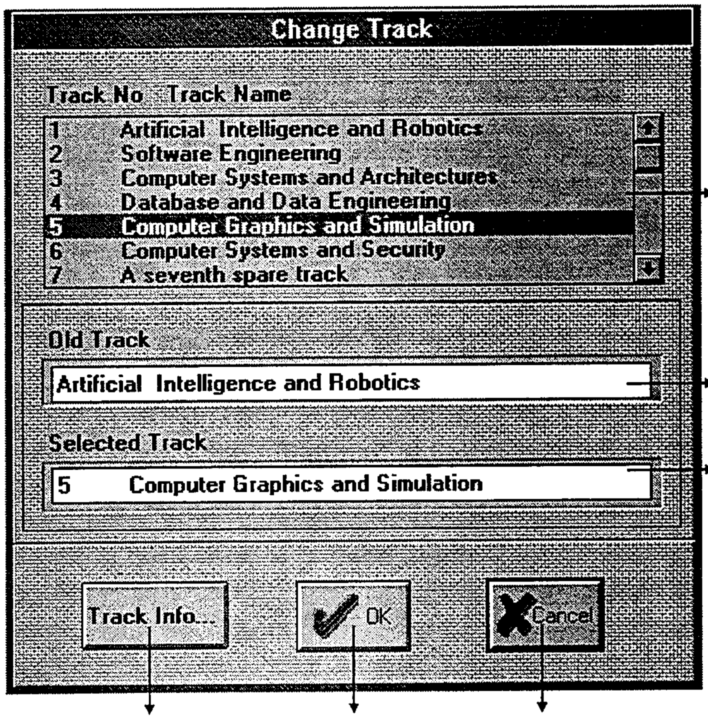
Closes the dialog box

Figure D18. Select Track Dialog Box

19. Change Track Dialog Box

If "Change Track" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D19 is displayed. In this dialog box, the user can do the following things:

- Getting more information about a track displayed in the list by first highlighting it and then pressing the "Track Info..." button.. The highlighted track name will be displayed in a text box just for a better feedback while the name of the old track will be displayed in another text box. If the "Track Info..." button is pressed, the information about the selected track will be displayed in "Track Info Dialog Box" shown in Figure D27.
- Selecting a track as the track of the student by first highlighting it and then pressing the "OK" button. There will be a confirmation question asked to the user whether he really wants to change the track of the student. When it is confirmed, only the track name of the selected track will be changed in student's record, but no change will be made in the student's matrix. The student will still be required to enter the required courses in the selected track manually. But the curriculum staff will not be asked to enter any course manually. For that reason, the curriculum staff's change of track is not recommended as long as it is not urgent.
- Canceling selecting a track and closing dialog box by pressing "Cancel" button.



Change Track

Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track

Old Track: Artificial Intelligence and Robotics

Selected Track: 5 Computer Graphics and Simulation

The list of valid tracks in the database

Old track

Selected track

Displays the track information about the selected track

Changes the track of the student to the newly selected track

Closes the dialog box

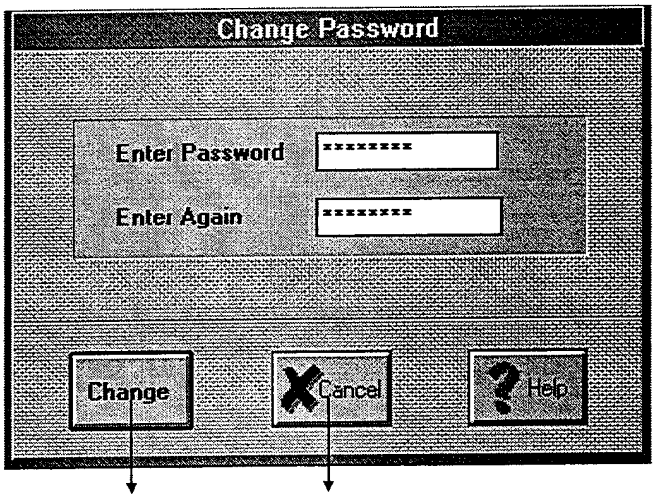
Figure D19. Change Track Dialog Box

20. Change Password Dialog Box

If "Change Password" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D20 is displayed. In this dialog box, the user can do the following things:

- Entering the new password twice and changing the old password of the student to the newly entered one by pressing the "Change" button. The new password has to be entered twice to make sure that the user is actually entering what he is intending to, since the entered password is not displayed on the screen. If the entered two passwords are not the same, the user is prompted to enter it again.
- Closing the dialog box without changing the password by pressing the "Cancel" button.
- Displaying the on-line help for "Change Password Dialog Box" by pressing the "Help" button.

This dialog box is basically designed for the student's use, although the curriculum staff can also use it.


Change Password

Enter Password *********

Enter Again *********

Changes the password of the student to the newly entered one

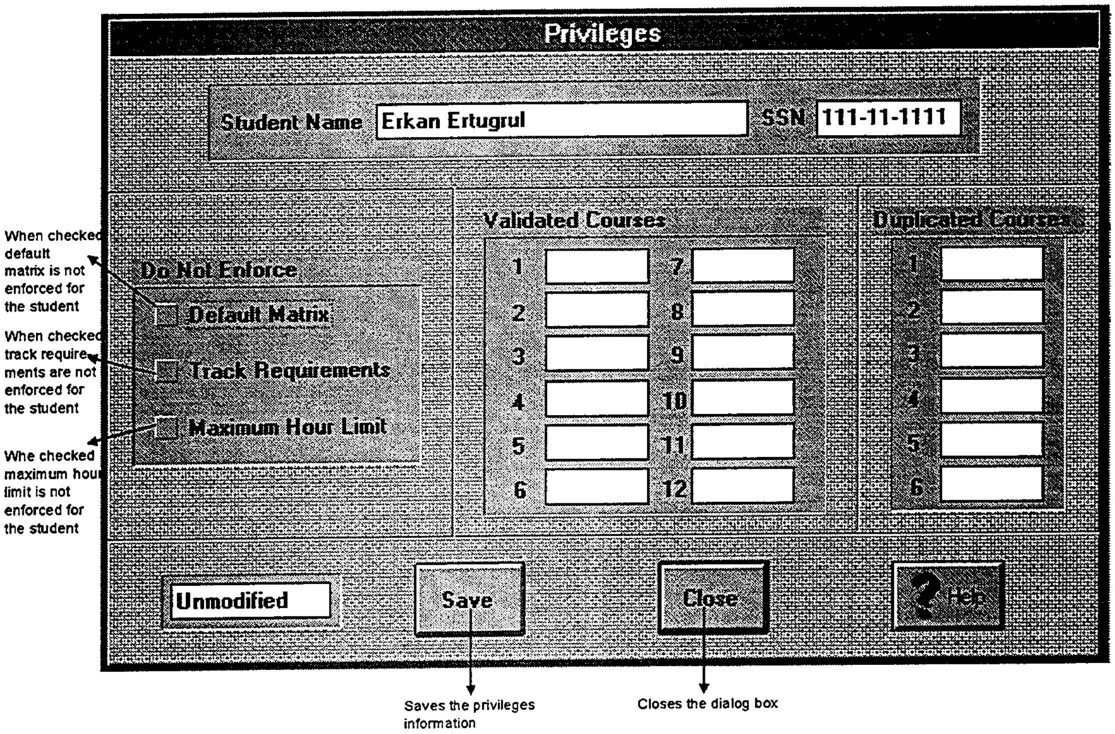
Closes the dialog box without changing the password

Figure D20. Change Password Dialog Box

21. Privileges Dialog Box

If "Privileges" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D21 is displayed. In this dialog box, the user can do the following things:

- Changing the privileges for the "Default Matrix", the " Track Requirements" and the "Maximum Hour Limit" by selecting or de-selecting the corresponding check boxes. If "Default Matrix" is checked, the student will be able to change the courses in his default matrix. If "Track Requirements" is checked, the student will not have to enter the requirements specified in a selected track. If "Maximum Hour Limit" is checked, the student will be able to enter courses whose credit hours' summation can exceed the maximum hour limit specified by the curricular officer.
- Entering the courses that the student validates. Thus the courses entered here will not be a requirement for the student to take if they are required courses.
- Entering the courses that the student is allowed to duplicate in his matrix. Normally a student is not allowed to duplicate a course except a few courses specified by the curricular officer in "Settings Dialog Box".
- Saving the changes made to the privileges by pressing the "Save" button.
- Closing the dialog box by pressing the "Close: button.
- Displaying the on-line help for "Privileges Dialog Box" by pressing the "Help" button.



Student Name Erkan Ertugrul

SSN 111-11-1111

□ Default Matrix

□ Track Requirements

□ Maximum Hour Limit

Unmodified

When checked default matrix is not enforced for the student

When checked track requirements are not enforced for the student

Whe checked maximum hour limit is not enforced for the student

Saves the privileges information

Closes the dialog box

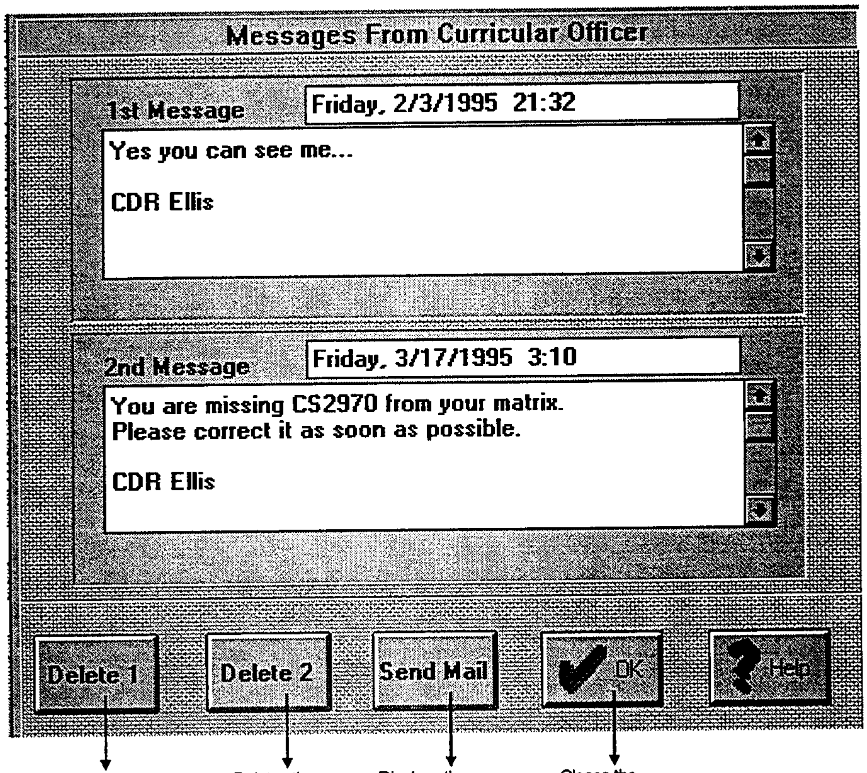
Figure D21. Privileges Dialog Box

22. Messages From Curricular Officer Dialog Pox

If "Mail Box" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D22 is displayed. In this dialog box, the user can do the following things:

- Deleting the displayed messages after reading them by pressing the "Delete 1" and "Delete 2" buttons.
- Displaying the "Send Mail Dialog Box" shown in Figure D23 to be able to send a message to the curricular officer by pressing the "Send Mail" button.
- Closing the dialog box by pressing the "OK" button.
- Displaying the on-line help for "Messages From Curricular Officer Dialog Box" by pressing the "Help" button.

This dialog box is basically designed for the student's use, although the curriculum staff can also use it.



1st Message Friday, 2/3/1995 21:32

Yes you can see me...

CDR Ellis

2nd Message Friday, 3/17/1995 3:10

You are missing CS2970 from your matrix.
Please correct it as soon as possible.

CDR Ellis

Deletes the first message

Deletes the second message

Displays the "Send Mail to Curriculum Officer" dialog box

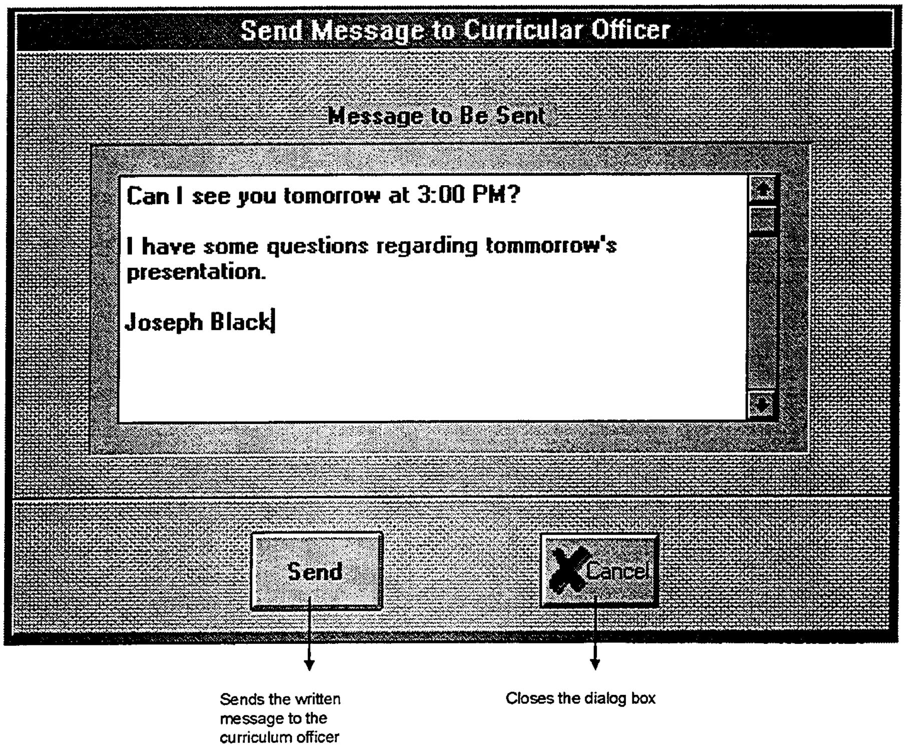
Closes the dialog box

Figure D22. Message From Curricular Officer Dialog Box

23. Send Message To Curricular Officer

If "Send Mail" command is selected from the "Messages From Curricular Officer Dialog Box", the dialog box shown in Figure D23 is displayed. In this dialog box, the user can do the following things:

- After writing the message in the space provided, sending it to the curricular officer by pressing the "Send" button.
- Closing the dialog box without sending anything by pressing the "Cancel" button.



Send Message to Curricular Officer
Message to Be Sent
Send
Cancel

Can I see you tomorrow at 3:00 PM?

I have some questions regarding tommorrow's presentation.

Joseph Black

Sends the written message to the curriculum officer
Closes the dialog box

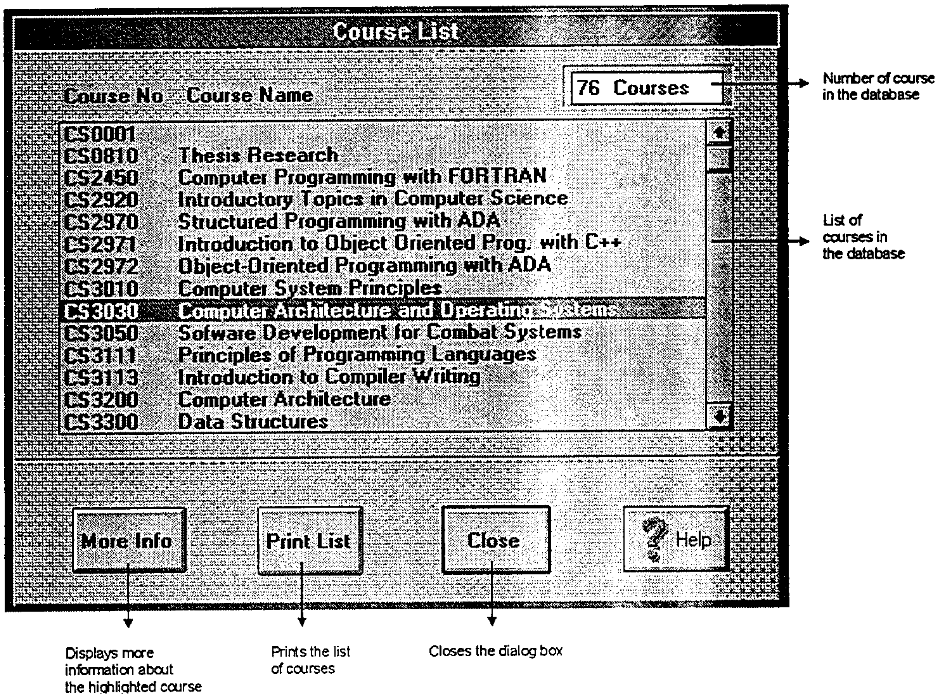
Figure D23. Send Message to Curricular Officer Dialog Box

24. Course List Dialog Box

If "Courses" command is selected from the "Student Matrix Dialog Box" or from the menu or the speed bar in the application window or from some other dialog box, the dialog box shown in Figure D24 is displayed. Once the list of the courses is displayed, the user can do the following things:

- Getting more information about the course highlighted in the list and displaying it in the "Course Info Dialog Box" shown in Figure D25 by pressing the "More Info" button.
- Printing the list of displayed courses by pressing the "Print List" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Course List Dialog Box" by pressing the "Help" button.

Double-clicking a course entry in the list will also bring more information about this course.



76 Courses

Course No	Course Name
CS0001	
CS0810	Thesis Research
CS2450	Computer Programming with FORTRAN
CS2920	Introductory Topics in Computer Science
CS2970	Structured Programming with ADA
CS2971	Introduction to Object Oriented Prog. with C++
CS2972	Object-Oriented Programming with ADA
CS3010	Computer System Principles
CS3030	Computer Architecture and Operating Systems
CS3050	Sofware Development for Combat Systems
CS3111	Principles of Programming Languages
CS3113	Introduction to Compiler Writing
CS3200	Computer Architecture
CS3300	Data Structures

Number of course in the database

List of courses in the database

Displays more information about the highlighted course

Prints the list of courses

Closes the dialog box

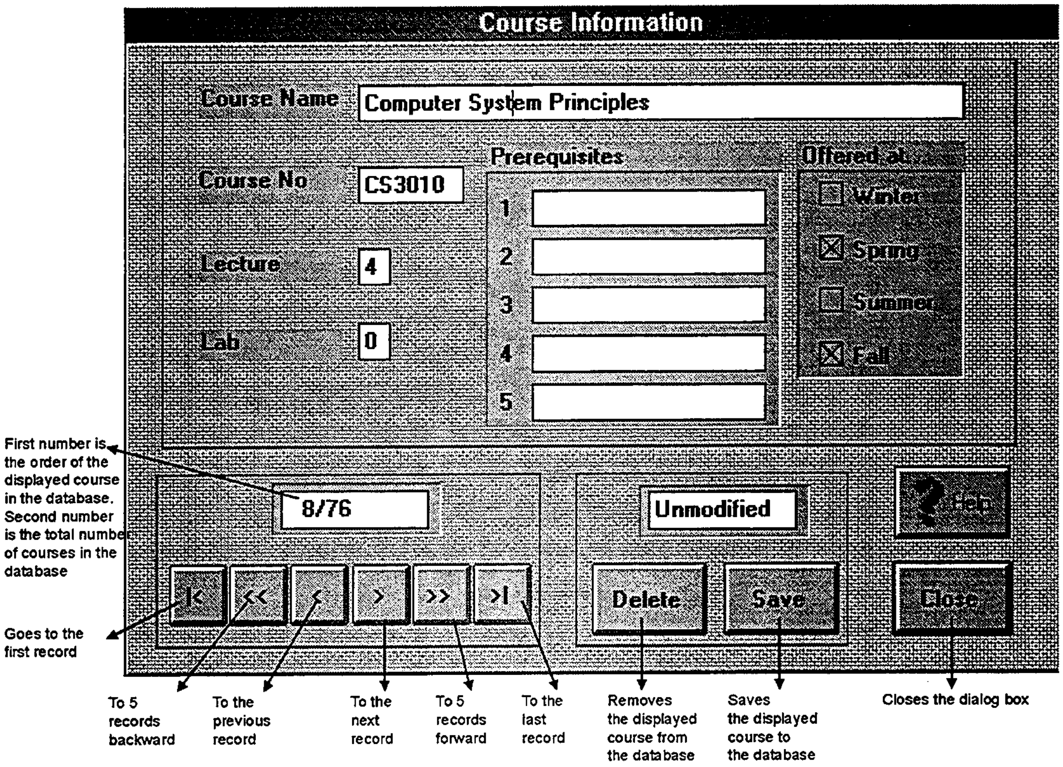
Figure D24. Course List Dialog Box

25. Course Information Dialog Box

If "More Info" command is selected from the "Student Matrix Dialog Box" or a course number is double-clicked in "Track Information Dialog Box", the dialog box shown in Figure D25 is displayed. Once the course information is displayed, the user can do the following things:

- Modifying the course information.
- Going to other course records by using the "Navigation Buttons".
- Deleting the displayed course from the database by pressing the "Delete" button.
- Saving the changes made to the course by pressing the "Save" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Course Information Dialog Box" by pressing the "Help" button.

It is important to note that, a course is not allowed to be deleted if it is a track requirement or a track elective for a track or a course in one of the default matrices. The user is first prompted to remove those courses from the tracks or default matrices found.



Course Information

Course Name: Computer System Principles

Course No: CS3010

Lecture: 4

Lab: 0

Prerequisites

1

2

3

4

5

Offered at

□ Winter

☒ Spring

□ Summer

☒ Fall

8/76

Unmodified

First number is the order of the displayed course in the database. Second number is the total number of courses in the database

Goes to the first record

To 5 records backward

To the previous record

To the next record

To 5 records forward

To the last record

Removes the displayed course from the database

Saves the displayed course to the database

Closes the dialog box

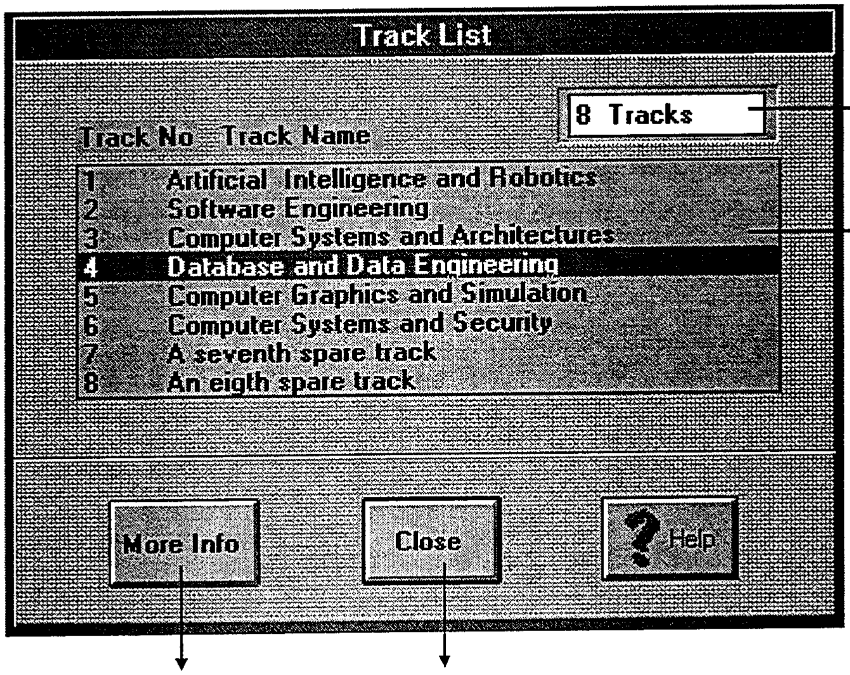
Figure D25. Course Information Dialog Box

26. Track List Dialog Box

If "Tracks" command is selected from the "Student Matrix Dialog Box" or from the menu or the speed bar in the application window or from some other dialog box, the dialog box shown in Figure D26 is displayed. Once the list of the tracks is displayed, the user can do the following things:

- Getting more information about the track highlighted in the list and displaying it in the "Track Info Dialog Box" shown in Figure D27 by pressing the "More Info" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Track List Dialog Box" by pressing the "Help" button.

Double-clicking a track entry in the list will also bring more information about this track.



8 Tracks

Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track
8	An eigth spare track

Number of tracks in the database

List of tracks in the database

Displays more information about the highlighted track

Closes the dialog box

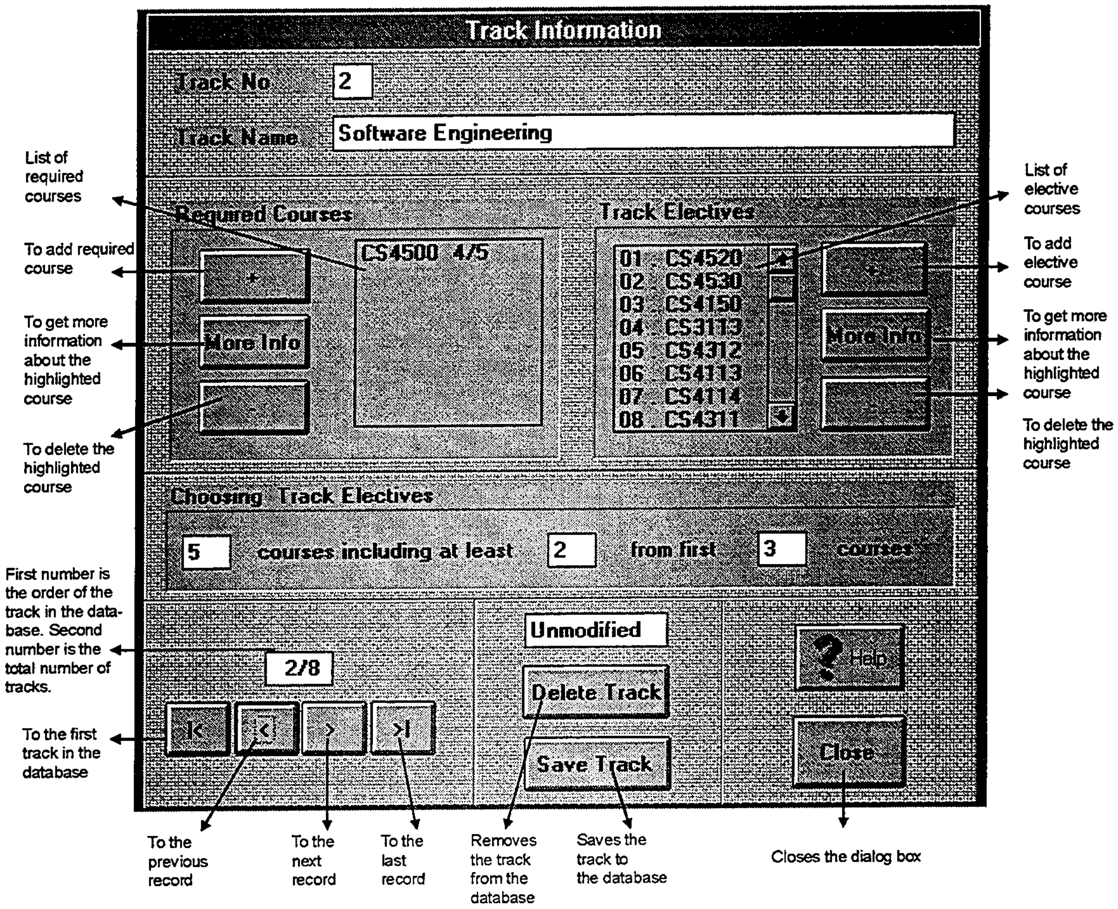
Figure D26. Track List Dialog Box

27. Track Information Dialog Box

If "More Info" command is selected from the "Student Matrix Dialog Box" or a track name is double-clicked in "Select Track" or "Change Track Dialog Box", the dialog box shown in Figure D27 is displayed. Once the track information is displayed, the user can do the following things:

- Modifying the track information.
- Adding required courses to the displayed track by pressing the "+" button in "Required Courses" section in the dialog box. The "Add Track Requirements Dialog Box" shown in Figure D28 is displayed to add those courses.
- Deleting a required course from the displayed track by pressing the "-" button in "Required Courses" section in the dialog box.
- Getting more information about a track requirement course by either pressing the "More Info button in "Required Courses" section or by double-clicking the course number in the "Required Courses" list.
- Adding elective courses to the displayed track by pressing the "+" button in "Elective Courses" section in the dialog box. The "Add Track Electives Dialog Box" shown in Figure D29 is displayed to add those courses.
- Deleting an elective course from the displayed track by pressing the "-" button in "Elective Courses" section in the dialog box.
- Getting more information about a track elective course by either pressing the "More Info button in "Elective Courses" section or by double-clicking the course number in the "Elective Courses" list.
- Going to other track records by using the "Navigation Buttons".
- Deleting the displayed track from the database by pressing the "Delete" button.
- Saving the changes made to the track by pressing the "Save" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Track Information Dialog Box" by pressing the "Help" button.

The rules of selecting the track electives are important to note for the students, because, if the students do not have the specified number of courses from the list of track electives in their matrices, they will not be allowed to save the matrices.



Track Information

Track No 2

Track Name Software Engineering

Required Courses

CS4500 4/5

Track Electives

01 CS4520
02 CS4530
03 CS4150
04 CS3113
05 CS4312
06 CS4113
07 CS4114
08 CS4311

Choosing Track Electives

5 courses including at least 2 from first 3 courses

2/8

Unmodified

List of required courses

To add required course

To get more information about the highlighted course

To delete the highlighted course

First number is the order of the track in the database. Second number is the total number of tracks.

To the first track in the database

List of elective courses

To add elective course

To get more information about the highlighted course

To delete the highlighted course

To the previous record

To the next record

To the last record

Removes the track from the database

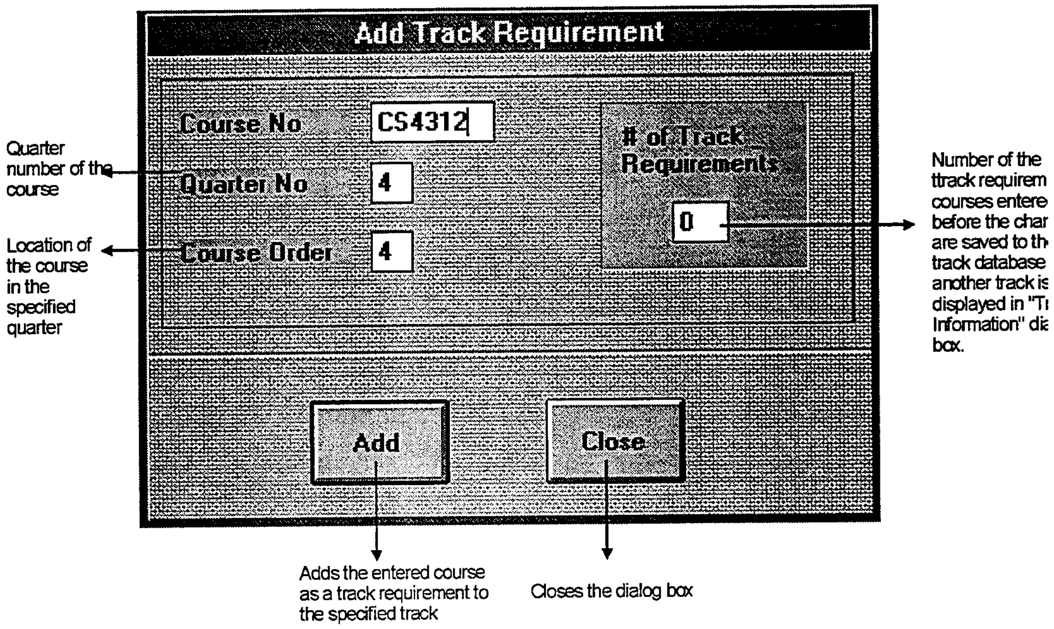
Saves the track to the database

Closes the dialog box

Figure D27. Track Information Dialog Box

28. Add Track Requirement Dialog Box

If "+" command for the "Track Requirements" is selected from the "Track Information Dialog Box", the dialog box shown in Figure D28 is displayed. In this dialog box, the user can enter track requirements as many as he wants without closing the dialog box. The number of track requirements entered before the changes are saved to the track database or another track is displayed in the "Track Information Dialog Box", are shown dynamically as the track requirement courses are entered one after another. Pressing the "Add" button will check whether the entered course number is valid and whether the "Quarter No" is between "1" and "8" and the "Course Order" is between "1" and "6". If everything is correct, the entered course is added to the list of "Track Requirements" and the edit boxes are cleared for another entry. When the "Close" button is pressed, the dialog box is closed and all of the required courses entered are immediately displayed in the "Track Requirements" section of the "Track Information Dialog Box". It is important however to note that the track requirements are not saved to the track database until the "Save" button in "Track Information Dialog Box" is pressed.



Add Track Requirement

Course No: CS4312

Quarter No: 4

Course Order: 4

# of Track Requirements: 0

Quarter number of the course

Location of the course in the specified quarter

Number of the ttrack requirem courses entere before the char are saved to th track database another track is displayed in "Tr Information" di box.

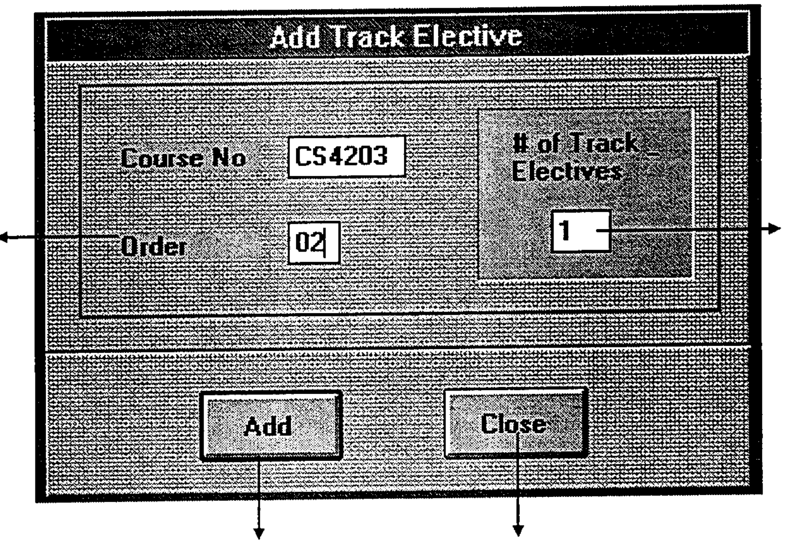
Adds the entered course as a track requirement to the specified track

Closes the dialog box

Figure D28. Add Track Requirement Dialog Box

29. Add Track Elective Dialog Box

If "+" command for the "Track Electives" is selected from the "Track Information Dialog Box", the dialog box shown in Figure D29 is displayed. In this dialog box, the user can enter track elective courses as many as he wants without closing the dialog box. The number of track electives that are entered before the changes are saved to the track database or another track is displayed in the "Track Information Dialog Box", are shown dynamically as the track elective courses are entered one after another. Pressing the "Add" button will check whether the entered course number is valid and whether the "Order" is entered. If everything is correct, the entered course is added to the list of "Track Electives" and the edit boxes are cleared for a new possible entry. When the "Close" button is pressed, the dialog box is closed and all of the required courses entered are immediately displayed in the "Track Electives" section of the "Track Information Dialog Box". It is important however to note that the track electives are not saved to the track database until the "Save" button in "Track Information Dialog Box" is pressed.



Add Track Elective

Course No: CS4203

Order: 02

# of Track Electives: 1

Specifies the order of the entered course in the Elective Courses List

Number of track elective courses entered before the changes are saved to the track databas or another track is displayed.

Adds the entered course as a track elective to the specified track

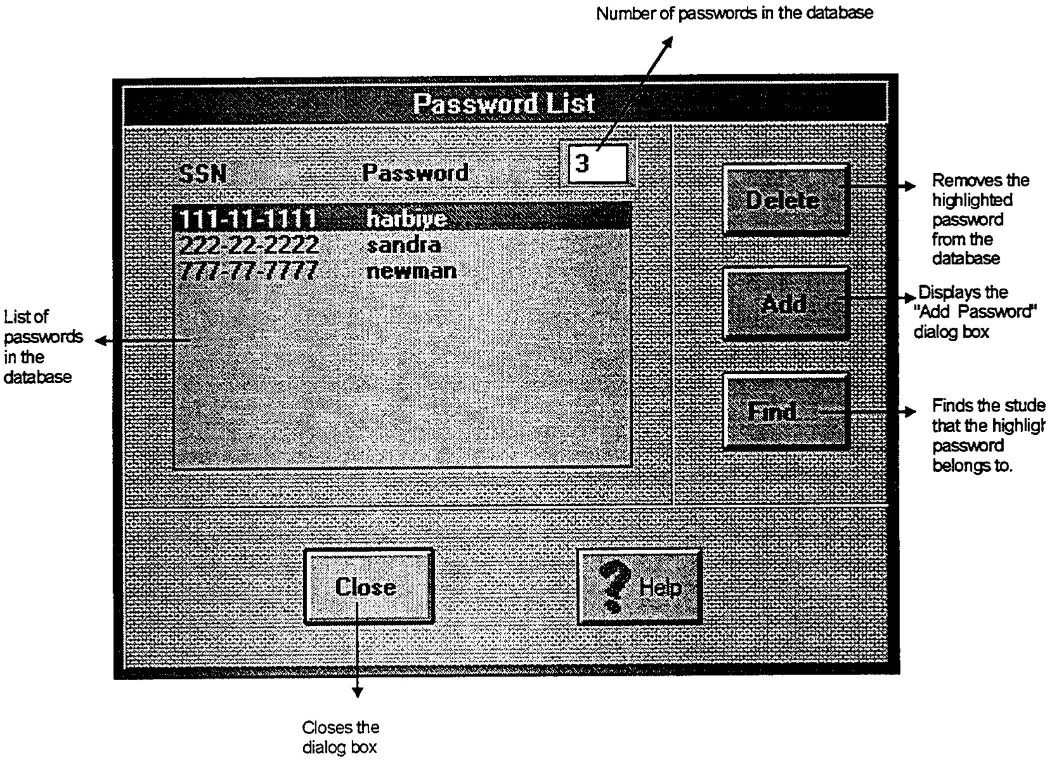
Closes the dialog box

Figure D29. Add Track Elective Dialog Box

30. Password List Dialog Box

If "Display Passwords" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D30 is displayed. Once the list of the passwords is displayed, the user can do the following things:

- Deleting a highlighted password entry from the database by pressing the "Delete" button.
- Adding a new password to the database by pressing the "Add" button. Pressing the "Add" button would display the "Add Password Dialog Box".
- Finding the student that a highlighted password belongs to by pressing the "Find..." button. The information belonging to the student found will be displayed in "Student Information Dialog Box". Double-clicking a password entry would also do the same job.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for "Password List Dialog Box".



SSN	Password
111-11-1111	harbiye
222-22-2222	sandra
777-77-7777	newman

3

Number of passwords in the database

Removes the highlighted password from the database

Displays the "Add Password" dialog box

Finds the stude that the highligt password belongs to.

List of passwords in the database

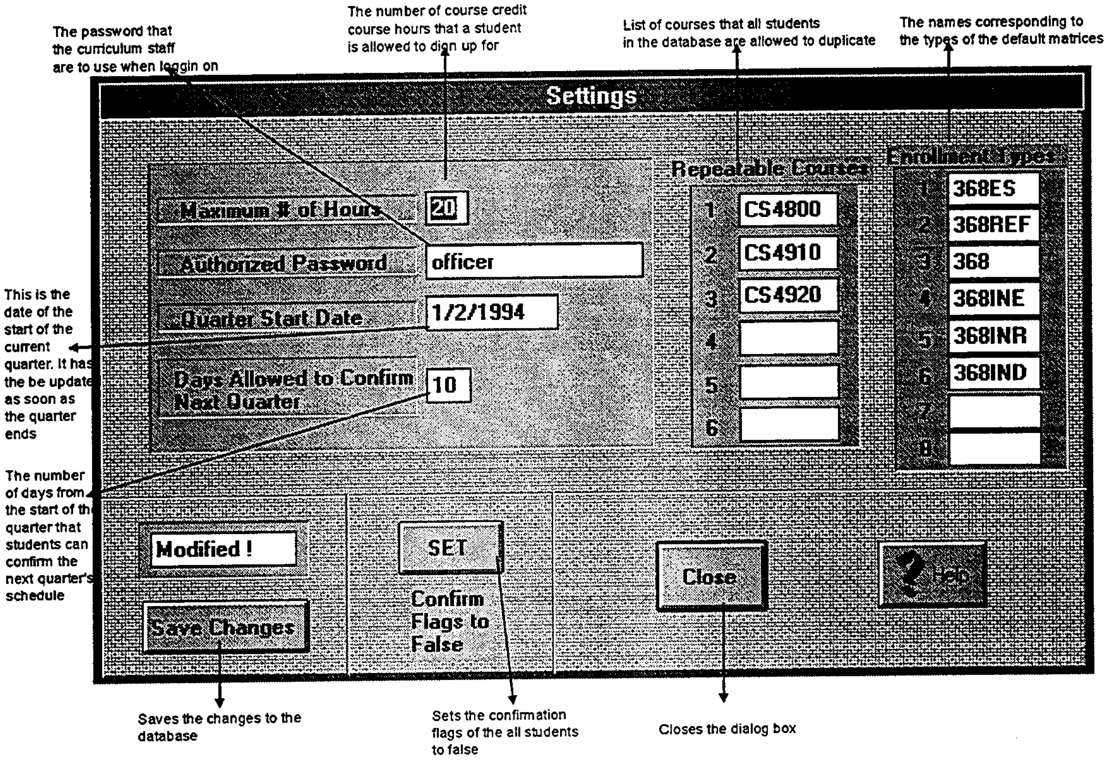
Closes the dialog box

Figure D30. Password List Dialog Box

31. Settings Dialog Box

If "Display Settings" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D31 is displayed. Once the settings information is displayed, the user can do the following things:

- Modifying "Maximum # of Hours". This specifies the maximum number of hours that a student can sign up for as courses if this student is not given the maximum hour privilege.
- Modifying "Authorized Password". This specifies the password that the curriculum staff will be using to get privileged access to the system.
- Modifying "Quarter Start Date". This information is important, because the period when the students are asked to confirm their next quarters' schedule is calculated by using this information and "Days Allowed to Confirm Next Quarter" information.
- Modifying "Days Allowed to Confirm Next Quarter". This is also used to calculate the period when the students are allowed to make their confirmations for the next quarter.
- Modifying the "Repeatable Courses". The repeatable courses are the courses that are allowed to be taken for more than once for all students.
- Modifying the "Enrollment Types". The "Enrollment Types" are the tags of the default matrices in the system. They show up as the labels that are explaining the kinds of the default matrices. The names can be up to 6 characters long.
- Saving the changes made to the settings information by pressing the "Save Changes" button.
- Setting all of the "Confirmation Flags" of the students to false by pressing the "Set Confirm Flags to False" button. The curricular officer may want to use this button before the next quarters' schedules are ready to be confirmed.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for the "Settings Dialog Box".



The password that the curriculum staff are to use when loggin on

The number of course credit course hours that a student is allowed to dign up for

List of courses that all students in the database are allowed to duplicate

The names corresponding to the types of the default matrices

This is the date of the start of the current quarter. It has the be update as soon as the quarter ends

The number of days from the start of th quarter that students can confirm the next quarter's schedule

Settings

Maximum # of Hours: 20

Authorized Password: officer

Quarter Start Date: 1/2/1994

Days Allowed to Confirm Next Quarter: 10

Repeatable Courses	
1	CS4800
2	CS4910
3	CS4920
4	
5	
6	

Enrollment Types	
1	368ES
2	368REF
3	368
4	368INE
5	368INR
6	368IND
7	
8	

Modified !

Saves the changes to the database

Sets the confirmation flags of the all students to false

Closes the dialog box

Figure D31. Settings Dialog Box

32. Pick Default Matrix Dialog Box

If "Display Default Matrix" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D32 is displayed. In this dialog box, the user can choose one of the displayed default matrices names and then click the corresponding button to see the default matrix desired in the "Default Matrix Dialog Box" shown in Figure D33. Pressing the "Cancel" button closes the dialog box without displaying any default matrix and pressing the "Help" button displays on-line help for "Pick Default Matrix Dialog Box".

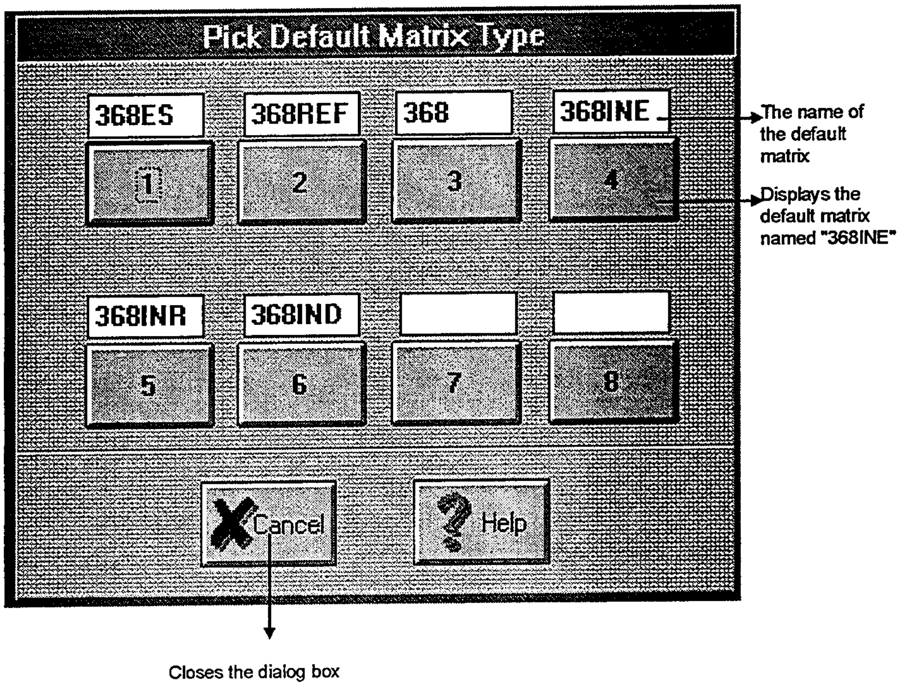


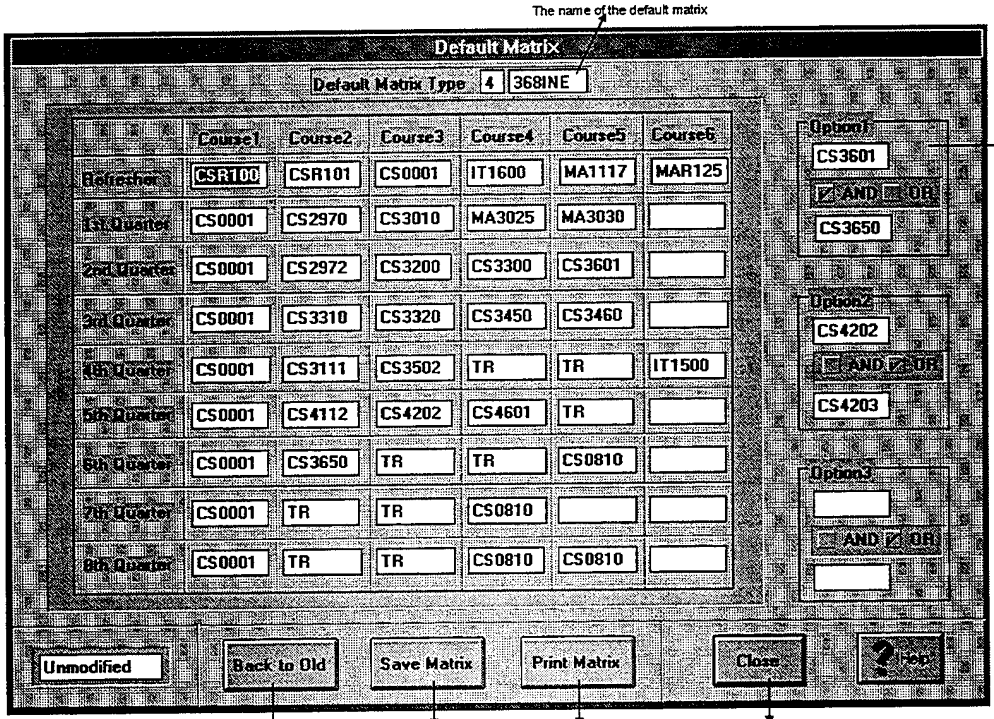
Figure D32. Pick Default Matrix Dialog Box

33. Default Matrix Dialog Box

When one of the default matrix option buttons is pressed from the "Pick Default Matrix Dialog Box" or "Default Matrix" button is pressed in "Student Matrix Dialog Box", the dialog box shown in Figure D33 is displayed. Once one of the default matrices is displayed, the user can do the following things:

- Modifying the courses in the matrix.
- Saving the changes made to the matrix by pressing the "Save" button.
- Undoing any changes made since the "Default Matrix Dialog Box is opened by pressing the "Back to Old" button.
- Printing the displayed default matrix by pressing the "Print Matrix" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for "Default Matrix Dialog Box" by pressing the "Help" button.

The name of the default matrix



Default Matrix Type 4 368INE

	Course1	Course2	Course3	Course4	Course5	Course6
Refresher	CSR100	CSR101	CS0001	IT1600	MA1117	MAR125
1st Quarter	CS0001	CS2970	CS3010	MA3025	MA3030	
2nd Quarter	CS0001	CS2972	CS3200	CS3300	CS3601	
3rd Quarter	CS0001	CS3310	CS3320	CS3450	CS3460	
4th Quarter	CS0001	CS3111	CS3502	TR	TR	IT1500
5th Quarter	CS0001	CS4112	CS4202	CS4601	TR	
6th Quarter	CS0001	CS3650	TR	TR	CS0810	
7th Quarter	CS0001	TR	TR	CS0810		
8th Quarter	CS0001	TR	TR	CS0810	CS0810	

Option1: CS3601 ☑ AND ☐ OR CS3650

Option2: CS4202 ☐ AND ☑ OR CS4203

Option3: ☐ AND ☑ OR

Unmodified

Two courses are displayed in each of the option boxes. If "AND" box is checked. that means, the student has to have both of the courses in his matrix. If "OR" box is checked that means, the student has to have at least one of these two courses in his matrix.

Displays the matrix that was displayed when this dialog box was opened by disregarding the changes

Saves the displayed matrix to the corresponding defaultmatrix file in the database

Prints the displayed default matrix

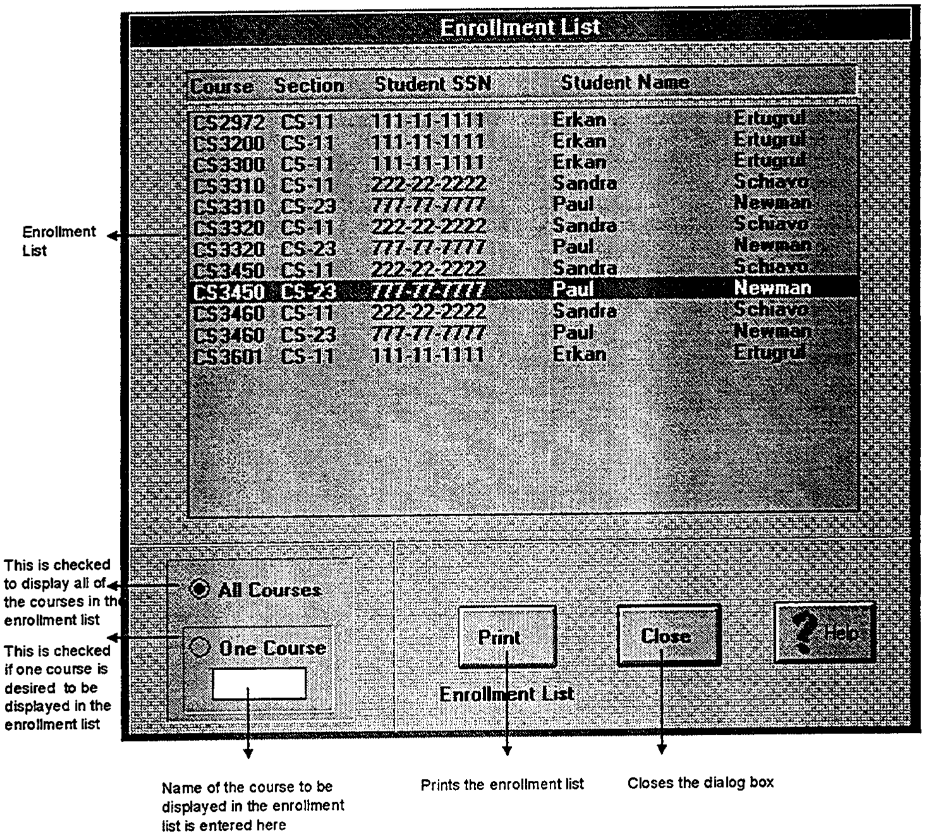
Closes the dialog box

Figure D33. Default Matrix Dialog Box

34. Enrollment List Dialog Box

If "Display Enrollment List" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D34 is displayed. Once the enrollment list is displayed, the user can do the following things:

- Displaying all of the courses enrolled by checking the "All Courses" radio button. By default, this option is checked when the dialog box is opened.
- Displaying one of the courses enrolled by first entering the name of the course to the edit box provided under the "One Course" radio button and later checking the "One Course" radio button.
- Printing the enrollment list by pressing the "Print Enrollment List" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for the "Enrollment List Dialog Box".



Course	Section	Student SSN	Student Name	
CS2972	CS-11	111-11-1111	Erkan	Ertugrul
CS3200	CS-11	111-11-1111	Erkan	Ertugrul
CS3300	CS-11	111-11-1111	Erkan	Ertugrul
CS3310	CS-11	222-22-2222	Sandra	Schiavo
CS3310	CS-23	777-77-7777	Paul	Newman
CS3320	CS-11	222-22-2222	Sandra	Schiavo
CS3320	CS-23	777-77-7777	Paul	Newman
CS3450	CS-11	222-22-2222	Sandra	Schiavo
CS3450	CS-23	777-77-7777	Paul	Newman
CS3460	CS-11	222-22-2222	Sandra	Schiavo
CS3460	CS-23	777-77-7777	Paul	Newman
CS3601	CS-11	111-11-1111	Erkan	Ertugrul

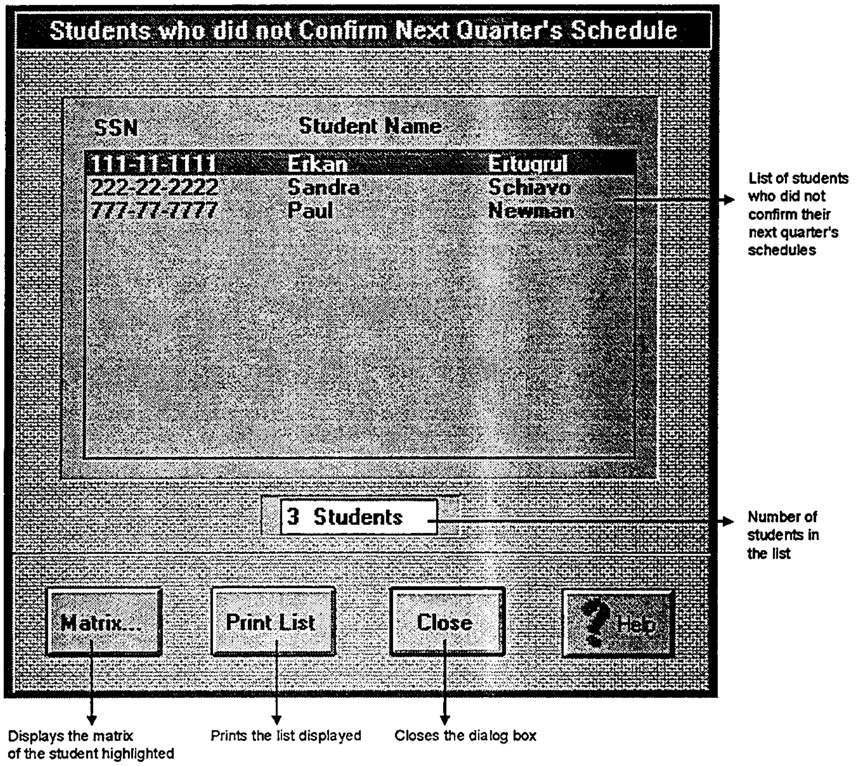
Enrollment List
Enrollment List
This is checked to display all of the courses in the enrollment list
This is checked if one course is desired to be displayed in the enrollment list
● All Courses
○ One Course
Print
Enrollment List
Close
? Help
Name of the course to be displayed in the enrollment list is entered here
Prints the enrollment list
Closes the dialog box

Figure D34. Enrollment List Dialog Box

35. Students Who Did Not Confirm Dialog Box

If "Display Ones Not Confirmed" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D35 is displayed. Once this list is displayed, the user can do the following things:

- Displaying the matrix of the highlighted student in "Student Matrix Dialog Box" by pressing the "Matrix" button. Double-clicking a student in the list will also bring the student's matrix.
- Printing the displayed list by pressing the "Print List" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for the "Students Who Did Not Confirm Dialog Box".



SSN	Student Name	
111-11-1111	Erkan	Ertugrul
222-22-2222	Sandra	Schiavo
777-77-7777	Paul	Newman

List of students who did not confirm their next quarter's schedules

Number of students in the list

Displays the matrix of the student highlighted

Prints the list displayed

Closes the dialog box

Figure D35. Students Who Did Not Confirm Dialog Box

36. Generating The Enrollment List

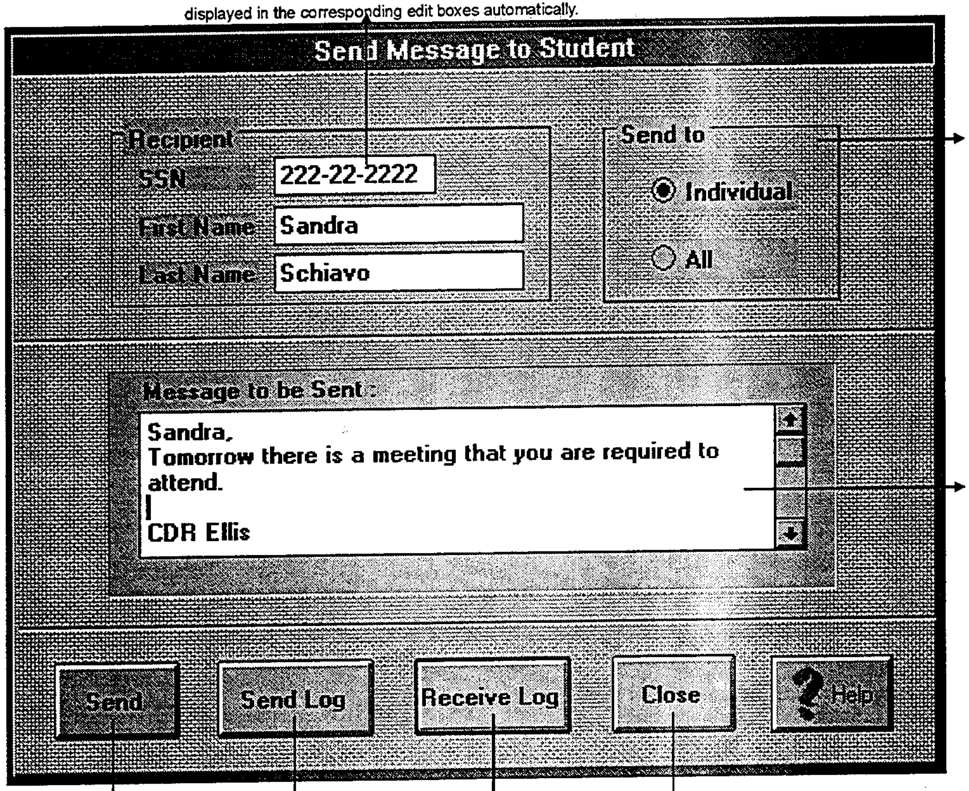
If "Generate Enrollment List" command is selected either from the application menu or the speed bar, the enrollment list is regenerated and the old enrollment list is deleted. A confirmation question is asked before the new enrollment list is generated. There is no harm in generating the enrollment list as many as the curriculum staff wants, since the regenerated enrollment list will reflect the latest matrices of the students. It is also very useful to use this tool even before the last day of the permitted time for the students to confirm their next quarters' schedules, since it will be possible to see the trend of enrolling for the courses. Of course the final generation of the enrollment list will be done after the students were not allowed to make changes in their next quarters' schedules. After a regeneration, the resulting enrollment list will be displayed in the "Enrollment List Dialog Box" shown in Figure D34.

37. Send Message To Student Dialog Box

If "Send Mail" command is selected either from the application menu or the speed bar or from the "Student List Dialog Box", the dialog box shown in Figure D36 is displayed. Once this list is displayed, the user can do the following things:

- Writing the SSN of the recipient. If the SSN is a valid one, the first and the last name of the student will be automatically displayed in the corresponding edit boxes as soon as the "SSN" edit box is left. If this dialog box is called from the " Student List Dialog Box", the recipient edit boxes will be filled with the information of the student highlighted in "Student List Dialog Box".
- Choosing to send a message to an individual by checking the "Individual" radio button. This is the default option when the dialog box is opened.
- Choosing to send a message to all of the students by checking the "All" radio button. When this option is selected, the text, if there is any, in "Recipient" section will be deleted and this section edit boxes will be "Read-only".
- Writing the message to be sent in the space provided.
- Sending the written message to the specified receivers by pressing the "Send" button.
- Displaying the "Send Log" shown in Figure D37, by pressing the "Send Log" button.
- Displaying the "Receive Log" shown in Figure D39, by pressing the "Receive Log" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for the "Send Message To Student Dialog Box".

The SSN of the recipient has toi be entered. If the entered SSN is valid, then as soon as the SSN edit box is left, the name of the recipient is displayed in the corresponding edit boxes automatically.



SSN 222-22-2222

First Name Sandra

Last Name Schiavo

Sandra,
Tomorrow there is a meeting that you are required to attend.

CDR Ellis

The message can either be sent to an individual or to all of the students

The message to be sent is entered here.

Sends the written message to the designated recipients

Displays the Send Log where the sent messages are kept until they are deleted.

Displays the Receive Log where the received messages are kept until they are deleted

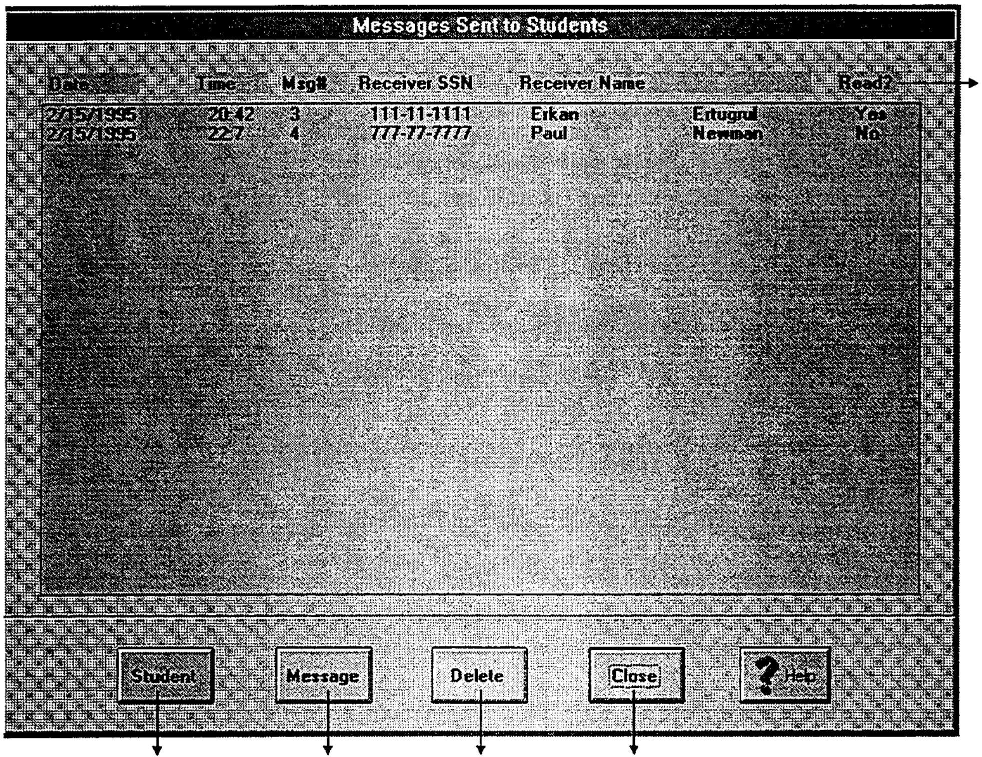
Closes the dialog box

Figure D36. Send Message To Student Dialog Box

38. Send Log Dialog Box

If "Send Log" button is pressed in "Send Message To Student Dialog Box", the dialog box shown in Figure D37 is displayed. Once the list of messages sent to the students is displayed, the user can do the following things:

- Getting more information about a highlighted student in the list by pressing the "Student" button. The found student information is displayed in "Student Information Dialog Box".
- Displaying the highlighted message in "Message Sent To Student Dialog Box" shown in Figure D38, by pressing the "Message" button. Double-clicking a message in the list will also show the sent message.
- Deleting a highlighted message from the list of sent messages by pressing the "Delete" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for the "Send Log Dialog Box".



Date	Time	Msg#	Receiver SSN	Receiver Name		Read?
2/15/1995	20:42	3	111-11-1111	Erkan	Ertugrul	Yes
2/15/1995	22:7	4	777-77-7777	Paul	Newman	No

Shows whether the student has read the sent message or not. The student is accepted as he has read the message as soon as he opens his mailbox.

Displays the information of the student highlighted

Displays the message whose entry is highlighted

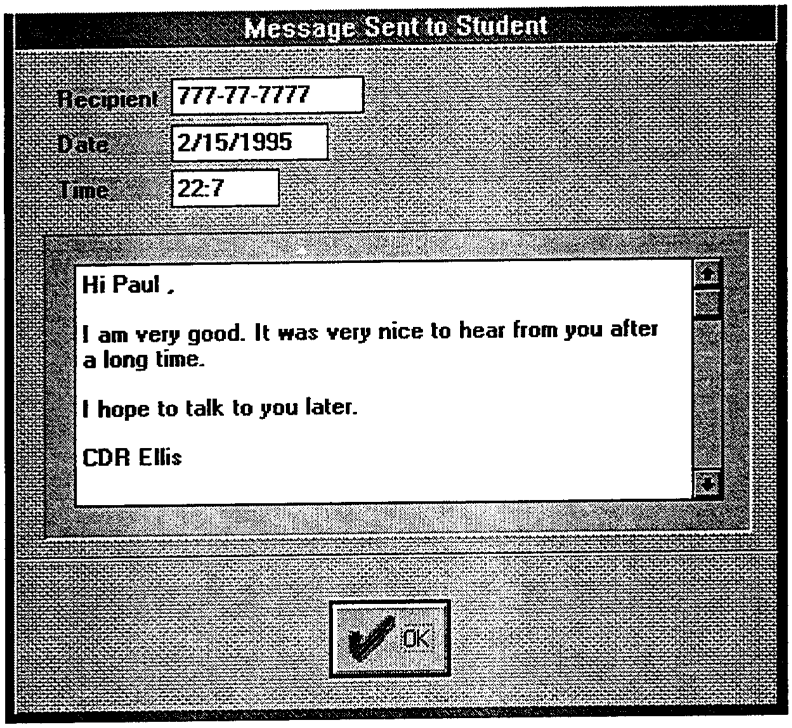
Deletes the highlighted message in the list

Closes the dialog box

Figure D37. Send Log Dialog Box

39. Message Sent To Student Dialog Box

If "Message" button in "Send Log Dialog Box" is pressed after a message item is highlighted in the list, the dialog box shown in Figure D38 is displayed. In this dialog box, the user can read the message sent to the student.



Recipient 777-77-7777

Date 2/15/1995

Time 22:7

Hi Paul ,

I am very good. It was very nice to hear from you after a long time.

I hope to talk to you later.

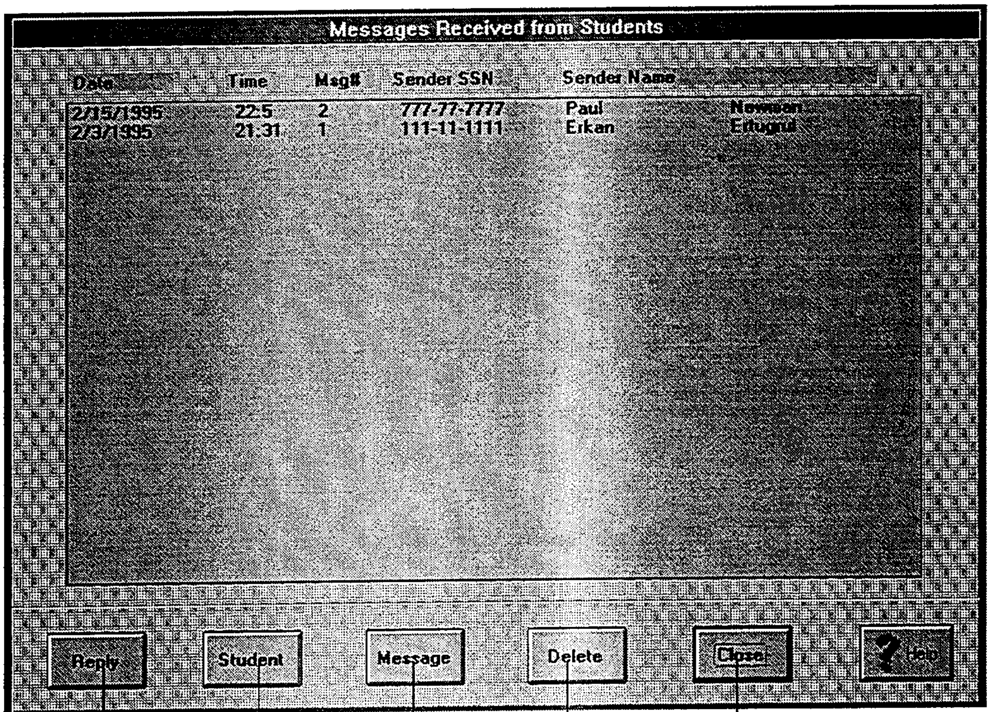
CDR Ellis

Figure D38. Message Sent To Student Dialog Box

40. Receive Log Dialog Box

If "Receive Log" button is pressed in "Receive Message To Student Dialog Box", the dialog box shown in Figure D39 is displayed. Once the list of messages received from the students is displayed, the user can do the following things:

- Replying a message received from a student by pressing the "Reply" button. The "Send Mail Dialog Box" will be displayed to write and send the mail.
- Getting more information about a highlighted student in the list by pressing the "Student" button. The found student information is displayed in "Student Information Dialog Box".
- Displaying the highlighted message in "Message Received from Student Dialog Box" shown in Figure D40, by pressing the "Message" button. Double-clicking a message in the list will also show the received message.
- Deleting a highlighted message from the list of received messages by pressing the "Delete" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for the "Receive Log Dialog Box".



Messages Received from Students

Date	Time	Msg#	Sender SSN	Sender Name	
2/15/1995	22:5	2	777-77-7777	Paul	Newman
2/3/1995	21:31	1	111-11-1111	Erkan	Ertugrul

Displays the "Send Mail" dialog box, with the highlighted message's sender is written as the recipient.

Displays the student information for the highlighted message's sender

Displays the the highlighted message's contents

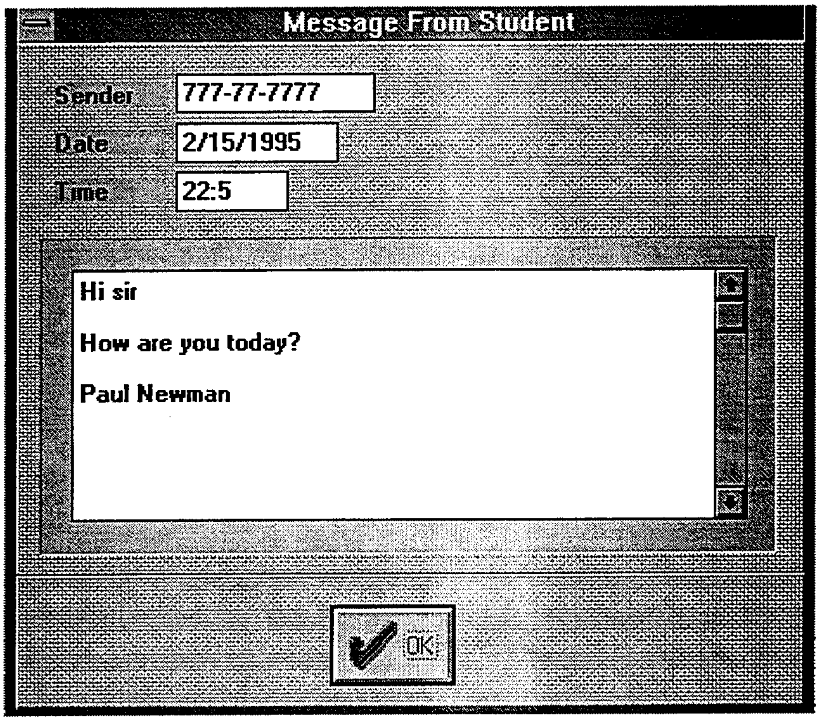
Deletes the highlighted message from the receive log

Closes the dialog box

Figure D39. Receive Log Dialog Box

41. Message From Student Dialog Box

If "Message" button in "Receive Log Dialog Box" is pressed after a message item is highlighted in the list, the dialog box shown in Figure D40 is displayed. In this dialog box, the user can read the message received from the student.



Sender 777-77-7777

Date 2/15/1995

Time 22:5

Hi sir

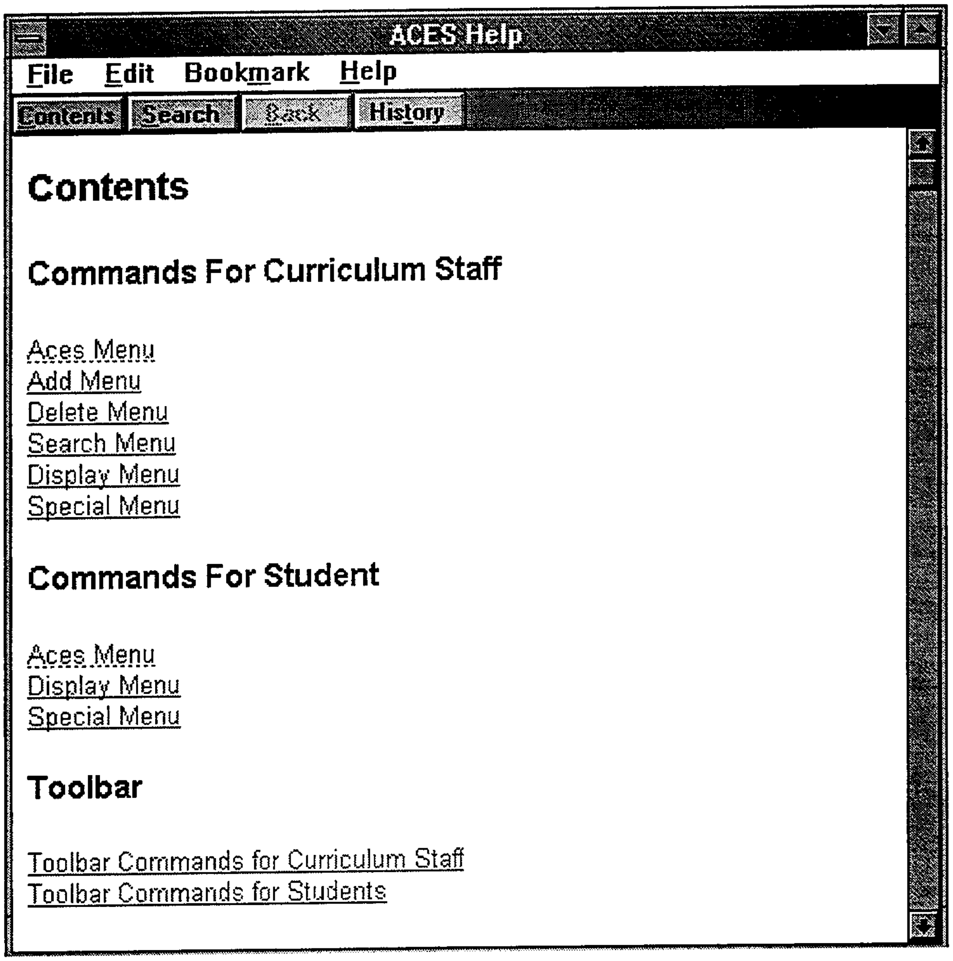
How are you today?

Paul Newman

Figure D40. Message From Student Dialog Box

42. On-Line Help

If "Help Contents" command is selected either from the application menu or the speed bar, the help window shown in Figure D41 is displayed. This help window is a typical on-line help for "Windows Environment". There are hyper links for the words or phrases written in green which means that clicking such a word or phrase will display another page with the new information about the selected word or phrase. It is possible to move back through the page links by pressing the "Back" button in the speed bar. For more information about how to use the on-line help, the "Using Help" command should be chosen from the application menu.



Contents

Commands For Curriculum Staff

Aces Menu
Add Menu
Delete Menu
Search Menu
Display Menu
Special Menu

Commands For Student

Aces Menu
Display Menu
Special Menu

Toolbar

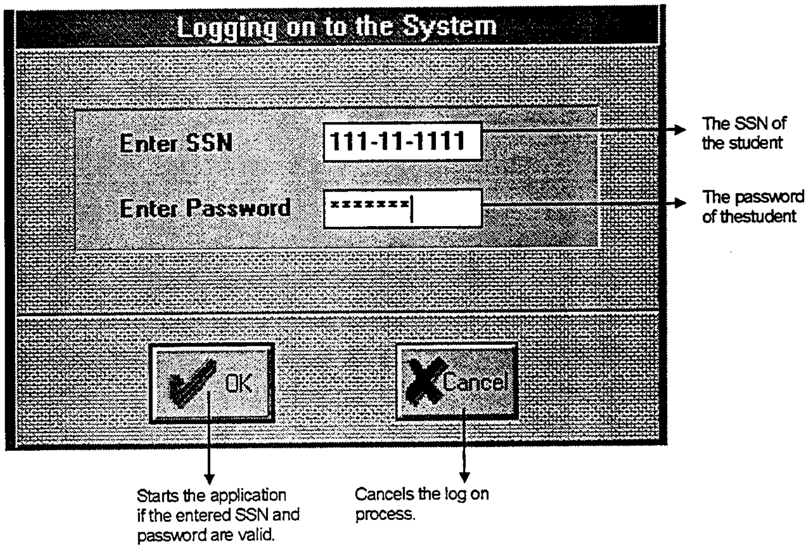
Toolbar Commands for Curriculum Staff
Toolbar Commands for Students

Figure D41. Aces Help Contents Window

D. USER MANUAL FOR THE STUDENTS

1. Logging On To The System

The students start the application by double clicking the application icon like any other applications in Windows Program Manager. The first dialog box which is shown in Figure D42 pops up in the center of the screen, waiting for the user to enter the "SSN" and the "Password". If the "SSN" and the "Password" are entered correctly, the "Student Matrix Dialog Box" shown in Figure D46, will be displayed for the student accessing to the system. When the student closes the "Student Matrix Dialog Box", the application window shown in Figure D44 is displayed.



Enter SSN 111-11-1111

Enter Password *******

Logging on to the System
OK
Cancel
The SSN of the student
The password of thestudent
Starts the application if the entered SSN and password are valid.
Cancels the log on process.

Figure D42. Logging On To The System Dialog Box

2. Application Window

When the "OK" button is pressed in "Log on Dialog Box", the application window shown in Figure D43 is displayed, if the password is correctly entered. In this application window, there is a menu that includes all of the functions that the students need to be able to interact with the system. A speedbar is also displayed right under the menu bar in order to provide easy access to the same functions in the menu. In the status bar, the description of either the menu commands or the speed bar commands are displayed dynamically as the mouse pointer moves over the speed bar or the menu commands.

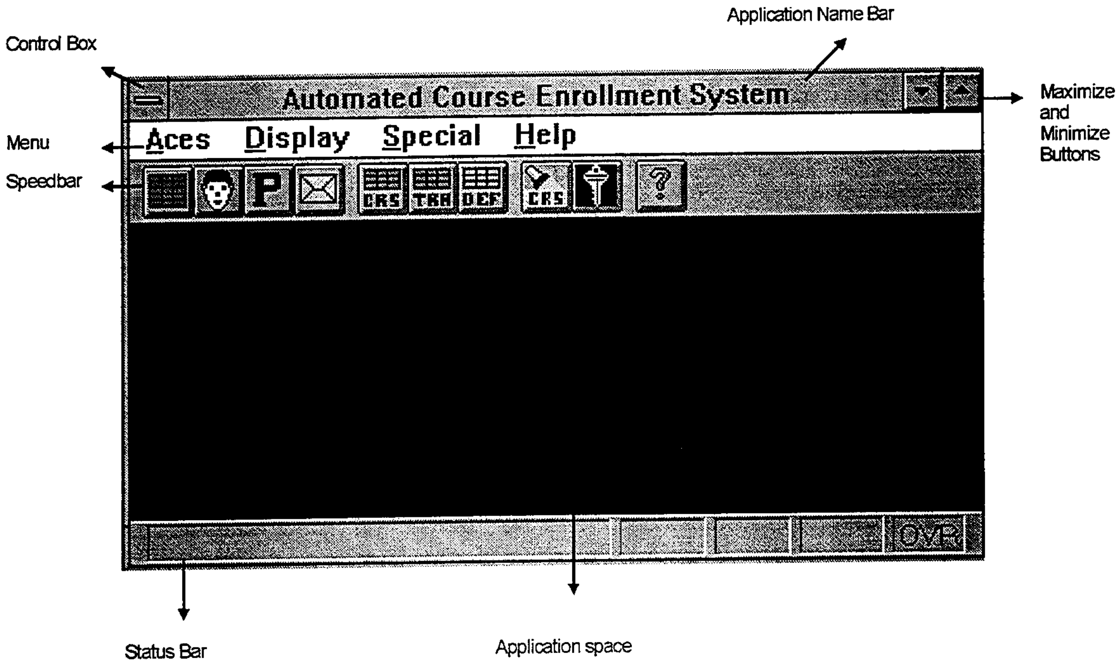

Application Name Bar
Control Box
Automated Course Enrollment System
Maximize and Minimize Buttons
Menu
Aces Display Special Help
Speedbar
P
CRS
TRN
DEF
CRS
?
OVR
Status Bar
Application space

Figure D43. Application Window For Students

3. Menu Commands

In Figure D44, the menu commands are displayed.

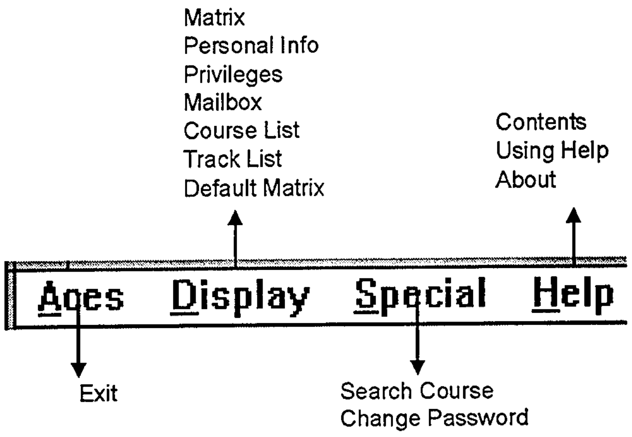


Figure D44. Menu Bar For Students

4. Speed Bar Commands

In Figure D45, Speed Bar commands are shown.

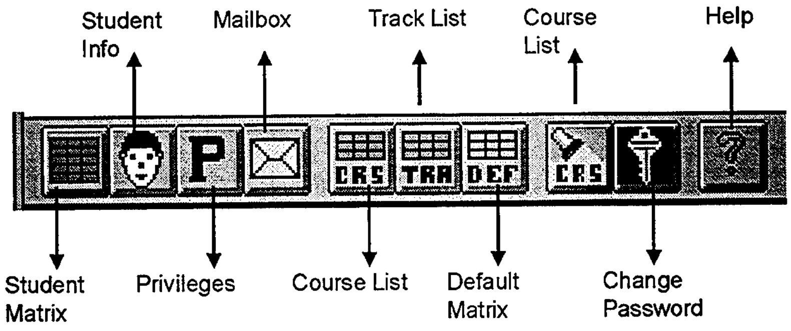
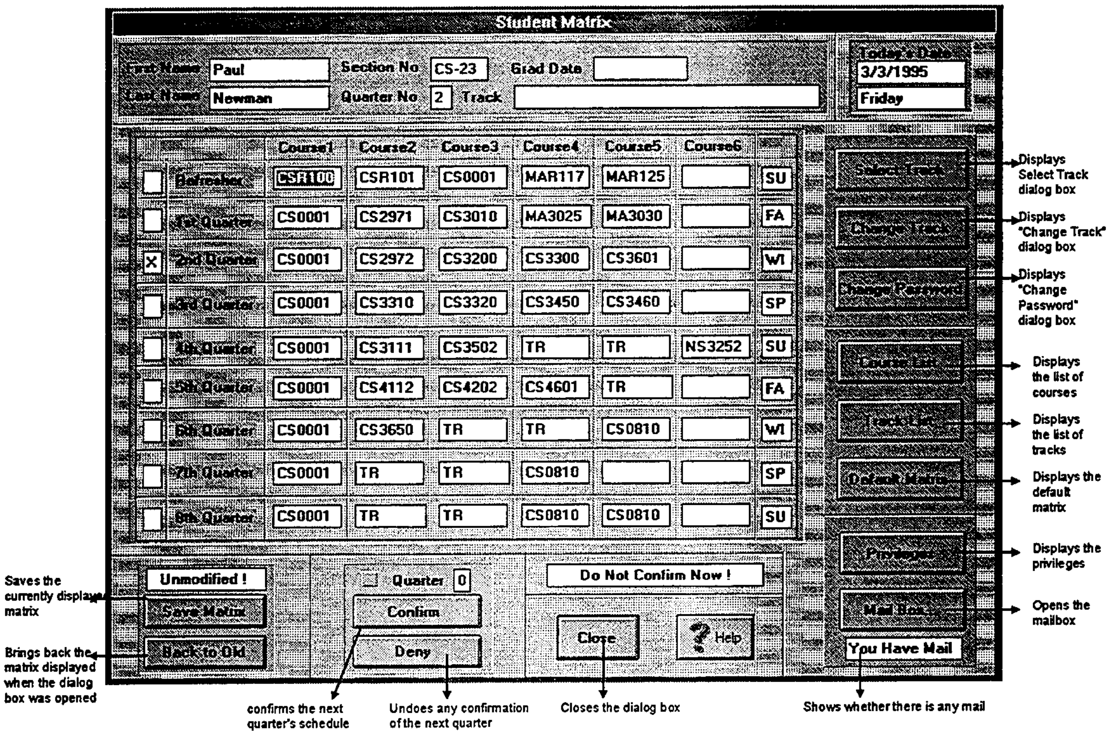


Figure D45. Speed Bar For Students

5. Student Matrix Dialog Box

If "Matrix..." command is selected from the "Student Information Dialog Box" or from the application menu or the speed bar, the dialog box shown in Figure D46 is displayed. Once the his matrix is displayed, the student can do the following things:

- Modifying the matrix.
- Selecting a track, if no track is selected yet, by pressing the "Select Track" button and then displaying the "Select Track Dialog Box" shown in Figure D47.
- Changing the track selected before, by pressing the "Change Track" button and then displaying the "Change Track Dialog Box" shown in Figure D48.
- Changing his password, by pressing the "Change Password" button and then displaying the "Change Password Dialog Box" shown in Figure D59.
- Displaying the course list in the "Course List Dialog Box" shown in Figure D53, by pressing the "Course List" button..
- Displaying the track list in the "Track List Dialog Box" shown in Figure D55, by pressing the "Track List" button.
- Displaying his default matrix in the "Default Matrix Dialog Box" shown in Figure D57, by pressing the "Default Matrix" button.
- Displaying the privileges given to him in "Privileges Dialog Box" shown in Figure D50, by pressing the "Privileges" dialog box
- Displaying his mailbox in "Student Mail Box" shown in Figure D51, by pressing the "Mail Box" button.
- Confirming his next quarter's schedule by pressing the "Confirm" button.
- Undoing the confirmation for the next quarter's schedule by pressing the "Deny" button.
- Saving the matrix by pressing the "Save" button.
- Bringing back the matrix that was first displayed when the "Student Matrix Dialog Box" was opened.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Student Matrix Dialog Box" by pressing the "Help" button.



First Name: Paul
Last Name: Newman
Section No: CS-23
Quarter No: 2
Grad Date:
Track:
Today's Date: 3/3/1995
Friday

		Course1	Course2	Course3	Course4	Course5	Course6	
□	Refresher	CSR100	CSR101	CS0001	MAR117	MAR125		SU
□	1st Quarter	CS0001	CS2971	CS3010	MA3025	MA3030		FA
☒	2nd Quarter	CS0001	CS2972	CS3200	CS3300	CS3601		WI
□	3rd Quarter	CS0001	CS3310	CS3320	CS3450	CS3460		SP
□	4th Quarter	CS0001	CS3111	CS3502	TR	TR	NS3252	SU
□	5th Quarter	CS0001	CS4112	CS4202	CS4601	TR		FA
□	6th Quarter	CS0001	CS3650	TR	TR	CS0810		WI
□	7th Quarter	CS0001	TR	TR	CS0810			SP
□	8th Quarter	CS0001	TR	TR	CS0810	CS0810		SU

Unmodified !

□ Quarter 0

Do Not Confirm Now !

You Have Mail

Displays Select Track dialog box

Displays "Change Track" dialog box

Displays "Change Password" dialog box

Displays the list of courses

Displays the list of tracks

Displays the default matrix

Displays the privileges

Opens the mailbox

Saves the currently displayed matrix

Brings back the matrix displayed when the dialog box was opened

confirms the next quarter's schedule

Undoes any confirmation of the next quarter

Closes the dialog box

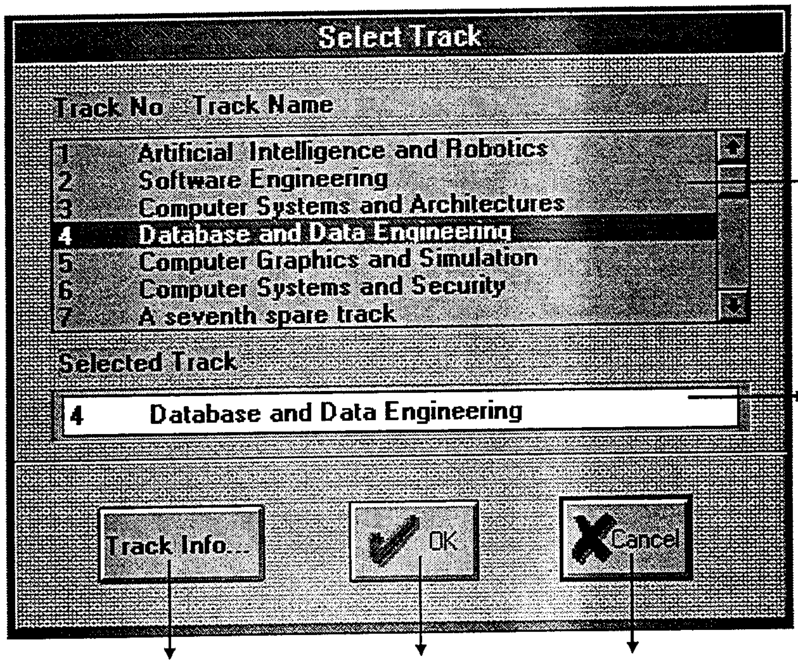
Shows whether there is any mail

Figure D46. Student Matrix Dialog Box

6. Select Track Dialog Box

If "Select Track" command is selected from the "Student Matrix Dialog Box" or from the application menu or the speed bar, the dialog box shown in Figure D47 is displayed. In this dialog box, the student can do the following things:

- Getting more information about a track displayed in the list by first highlighting it and then pressing the "Track Info..." button.. The highlighted track name will be displayed in another text box just for a better feedback and if the "Track Info..." button is pressed, the information about the selected track will be displayed in "Track Info Dialog Box" shown in Figure D56.
- Selecting a track as his track by first highlighting it and then pressing the "OK" button. There will be a confirmation question asked to the student. When it is confirmed, the track requirement courses in the selected track will be automatically copied to the student's matrix. The elective courses of the track will still have to be entered manually.
- Canceling selecting a track and closing dialog box by pressing "Cancel" button.



Select Track

Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track

Selected Track

4 Database and Data Engineering

The list of valid tracks in the datab

The selected track

Displays the track information for the selected track

The student's track becomes saved as the selected track here

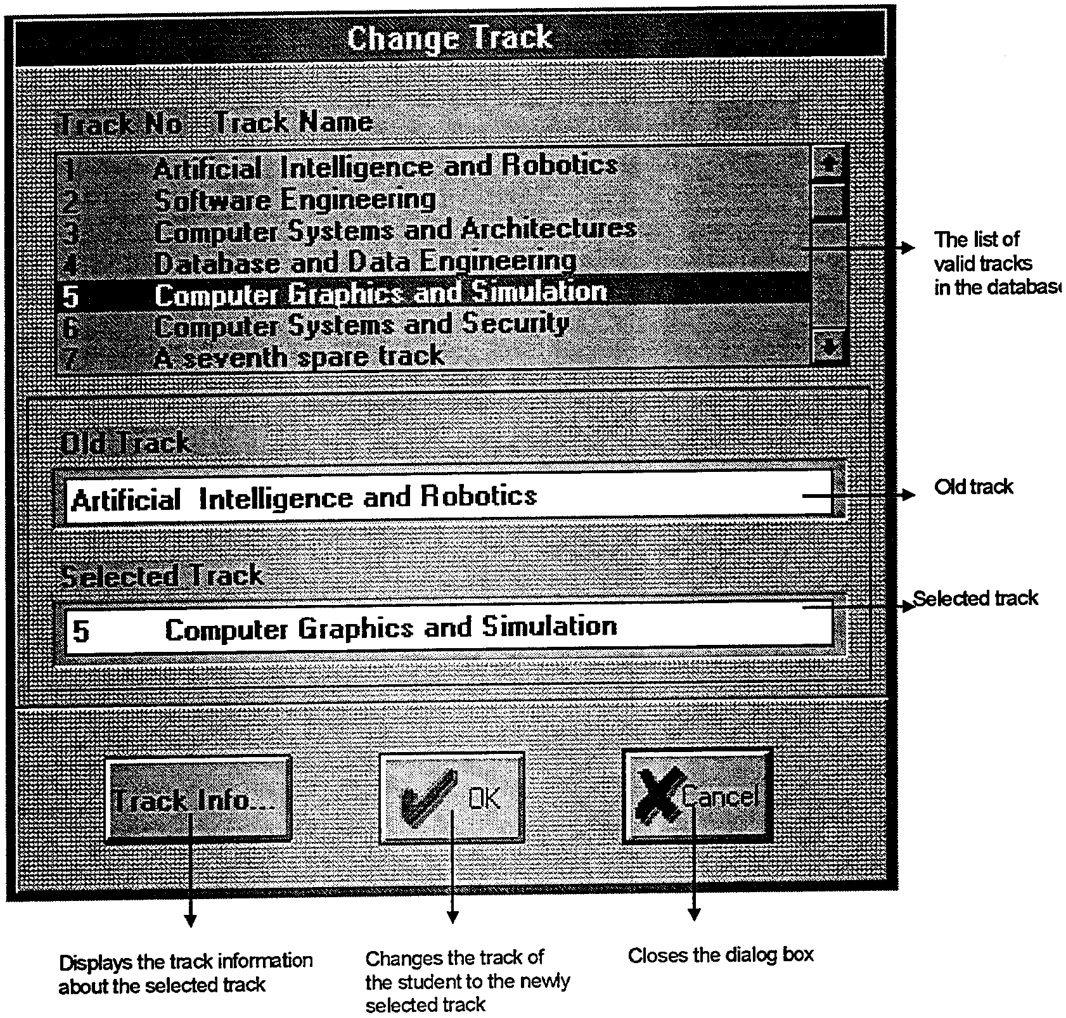
Closes the dialog box

Figure D47. Select Track Dialog Box

7. Change Track Dialog Box

If "Change Track" command is selected from the "Student Matrix Dialog Box" or from the application menu or the speed bar, the dialog box shown in Figure D48 is displayed. In this dialog box, the student can do the following things:

- Getting more information about a track displayed in the list by first highlighting it and then pressing the "Track Info..." button.. The highlighted track name will be displayed in a text box just for a better feedback while the name of the old track will be displayed in another text box. If the "Track Info..." button is pressed, the information about the selected track will be displayed in "Track Info Dialog Box" shown in Figure D56.
- Selecting a track as his track by first highlighting it and then pressing the "OK" button. There will be a confirmation question asked to the user whether he really wants to change the track of the student. When it is confirmed, only the track name of the selected track will be changed in student's record, but no change will be made in the student's matrix. He will still be required to enter the required courses in the selected track manually.
- Canceling selecting a track and closing dialog box by pressing "Cancel" button.



Change Track

Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track

Old Track

Artificial Intelligence and Robotics

Selected Track

5 Computer Graphics and Simulation

The list of valid tracks in the databas

Old track

Selected track

Displays the track information about the selected track

Changes the track of the student to the newly selected track

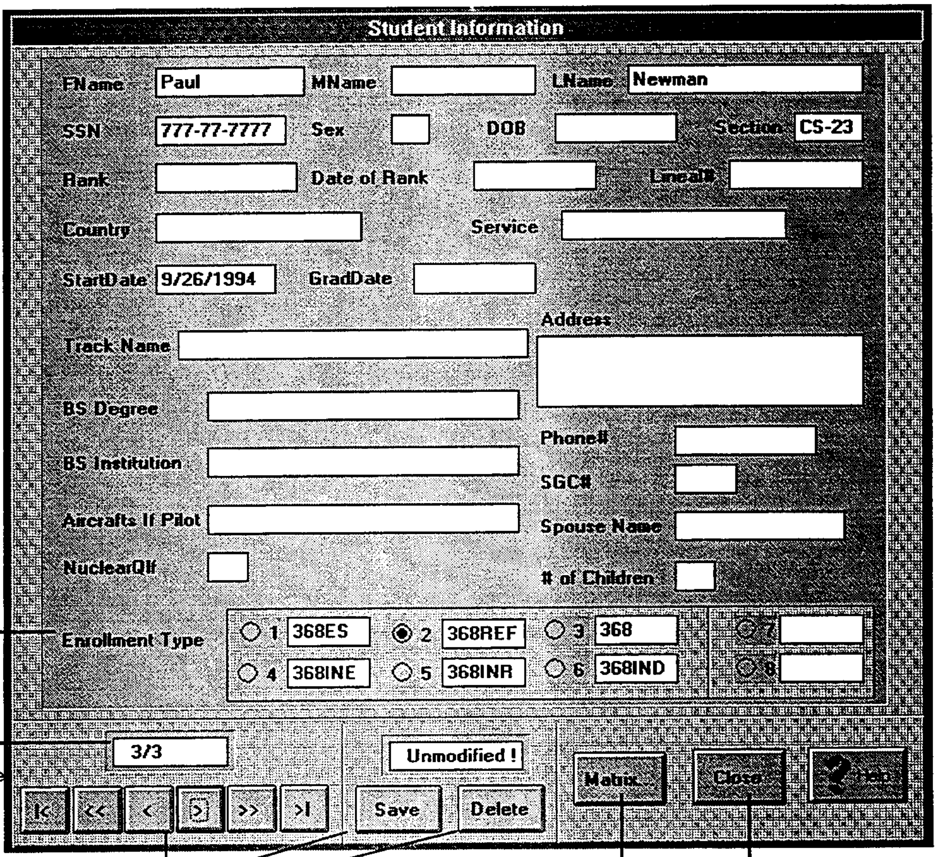
Closes the dialog box

Figure D48. Change Track Dialog Box

8. Student Information Dialog Box

If "Student Info..." command is selected from the application menu or the speed bar, the dialog box shown in Figure D49 is displayed. Once his information is displayed, the student can do the following things:

- Displaying his matrix in "Student Matrix Dialog Box" shown in Figure D46 by pressing the "Matrix" button.
- Closing the dialog box by pressing the Close button.
- Displaying the on-line help for "Student Information Dialog Box" by pressing the "Help" button



Student Information

FName Paul MName LName Newman

SSN 777-77-7777 Sex DOB Section CS-23

Rank Date of Rank Lineal#

Country Service

StartDate 9/26/1994 GradDate

Address

Track Name

BS Degree

Phone#

BS Institution

SGC#

Aircrafts If Pilot

Spouse Name

NuclearQlf

# of Children

Enrollment Type ○ 1 368ES ● 2 368REF ○ 3 368 ○ 7
○ 4 368INE ○ 5 368INR ○ 6 368IND ○ 8

3/3

Unmodified !

Shows the type of the default matrix that the student is assigned

First number is the order number of the student in the database Second is the total number of students in the database

These buttons are not functional

Displays the matrix of the student

Closes the dialog box

Figure D49. Student Information Dialog Box

9. Privileges Dialog Box

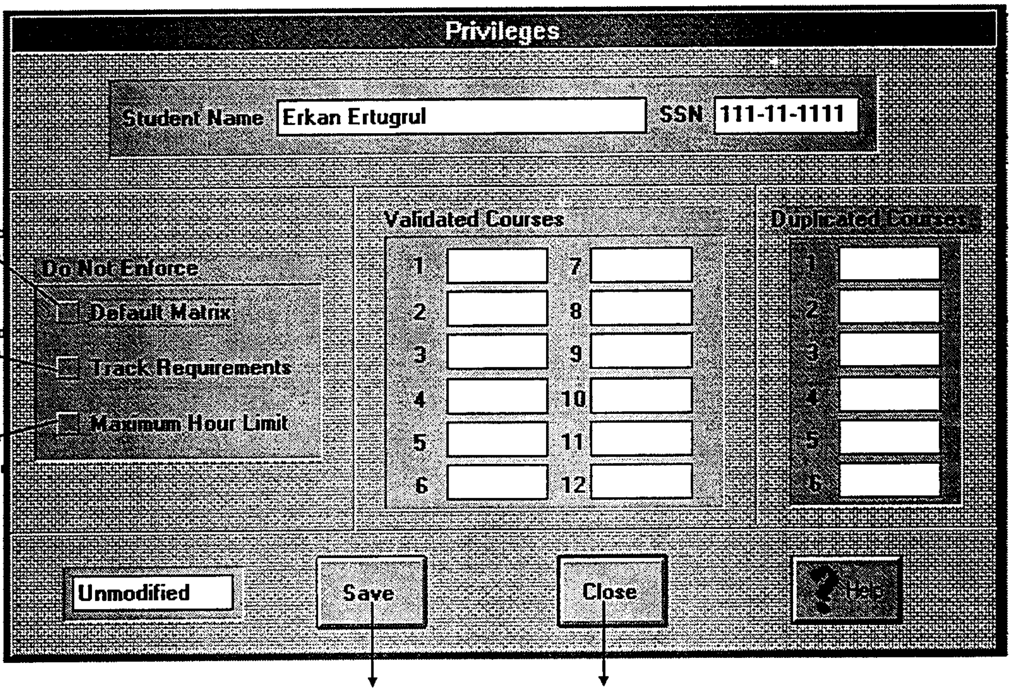
If "Privileges" command is selected from the "Student Matrix Dialog Box" or from the application menu or the speed bar, the dialog box shown in Figure D50 is displayed. In this dialog box, the user can do the following things:

- Getting information about the privileges. If "Default Matrix" is checked, he will be able to change the courses in his default matrix. If "Track Requirements" is checked, he will not have to enter the requirements specified in a selected track. If "Maximum Hour Limit" is checked, he will be able to enter courses whose credit hours' summation can exceed the maximum hour limit put by the curricular officer.
- Getting information about the validated courses. The courses entered here will not be a requirement for him to take if they are required courses. He is not allowed to enter validated courses. This has to be done by the curriculum staff.
- Getting information about the courses that he is allowed to duplicate in his matrix. Normally a student is not allowed to duplicate a course except a few courses specified by the curricular officer in "Settings Dialog Box". He is not allowed to enter these courses. This has to be done by the curriculum staff.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Privileges Dialog Box" by pressing the "Help" button.

When checked default matrix is not enforced for the student

When checked track requirements are not enforced for the student

Whe checked maximum hour limit is not enforced for the student



Student Name: Erkan Ertugrul

SSN: 111-11-1111

□ Default Matrix
□ Track Requirements
□ Maximum Hour Limit

Unmodified

This button is not functional

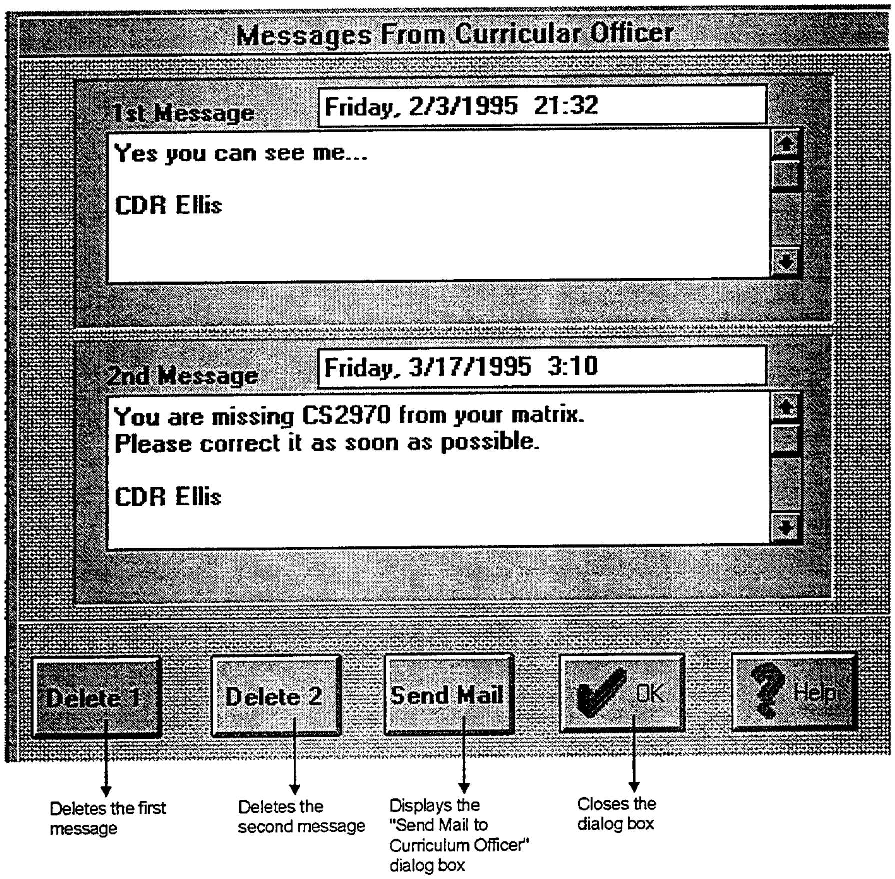
Closes the dialog box

Figure D50. Privileges Dialog Box

10. Messages From Curricular Officer Dialog Box

If "Mail Box" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D51 is displayed. In this dialog box, the student can do the following things:

- Deleting the displayed messages after reading them by pressing the "Delete 1" and "Delete 2" buttons.
- Displaying the "Send Mail Dialog Box" shown in Figure D52 to be able to send a message to the curricular officer by pressing the "Send Mail" button.
- Closing the dialog box by pressing the "OK" button.
- Displaying the on-line help for "Messages From Curricular Officer Dialog Box" by pressing the "Help" button.



Messages From Curricular Officer

1st Message: Friday, 2/3/1995 21:32

Yes you can see me...

CDR Ellis

2nd Message: Friday, 3/17/1995 3:10

You are missing CS2970 from your matrix.
Please correct it as soon as possible.

CDR Ellis

Deletes the first message

Deletes the second message

Displays the "Send Mail to Curriculum Officer" dialog box

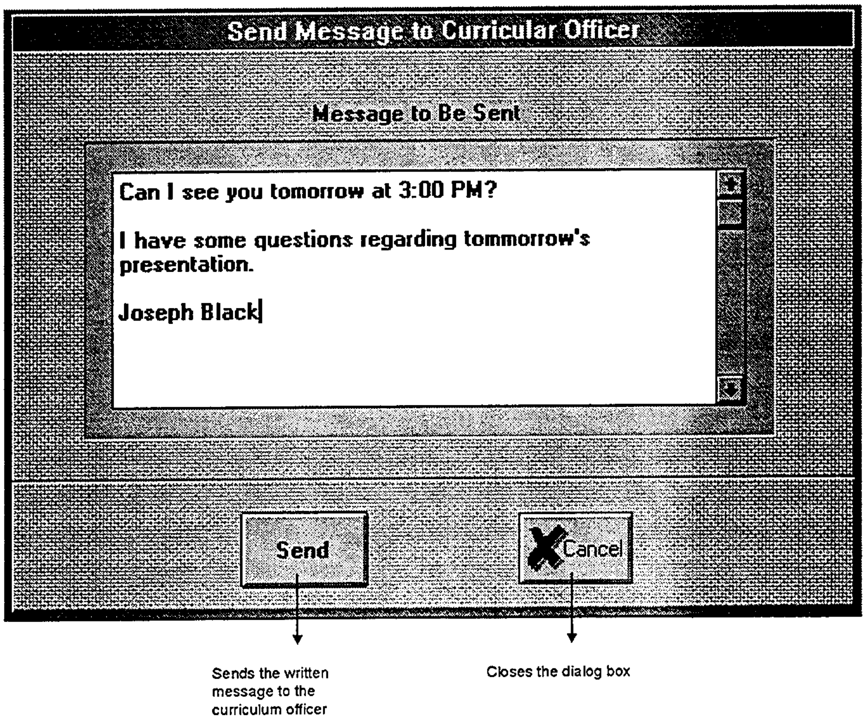
Closes the dialog box

Figure D51. Messages From Curricular Officer Dialog Box

11. Send Message To Curricular Officer Dialog Box

If "Send Mail" command is selected from the "Messages From Curricular Officer Dialog Box", the dialog box shown in Figure D52 is displayed. In this dialog box, the student can do the following things:

- After writing the message in the space provided, sending it to the curricular officer by pressing the "Send" button.
- Closing the dialog box without sending anything by pressing the "Cancel" button.



Can I see you tomorrow at 3:00 PM?

I have some questions regarding tommorrow's presentation.

Joseph Black

Sends the written message to the curriculum officer

Closes the dialog box

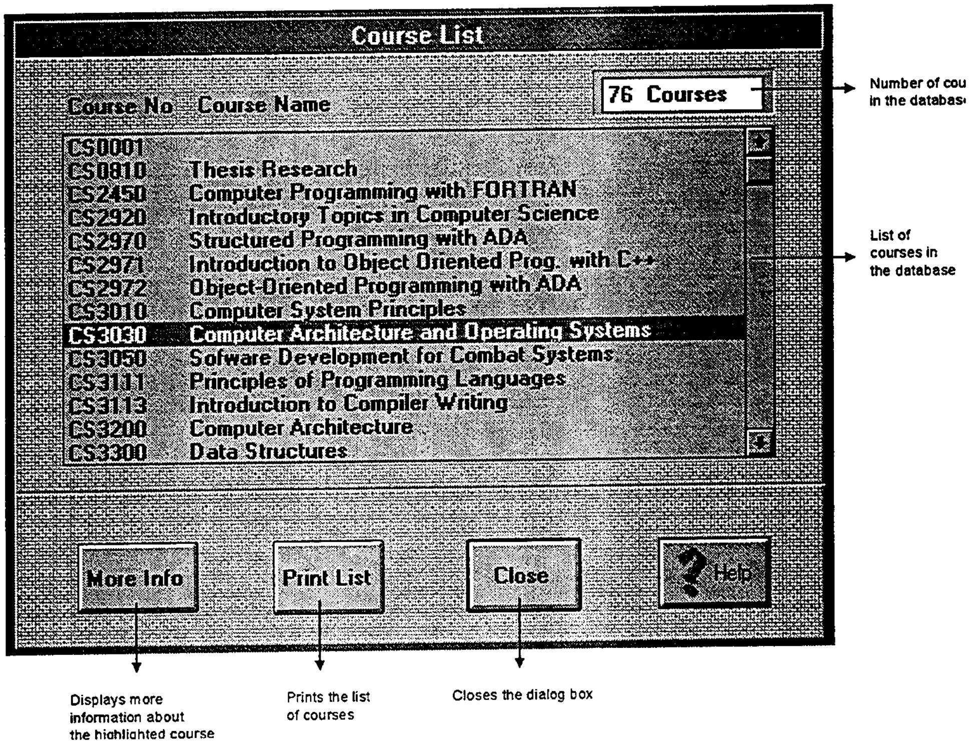
Figure D52. Send Message to Curricular Officer Dialog Box

12. Course List Dialog Box

If "Courses" command is selected from the "Student Matrix Dialog Box" or from the menu or the speed bar in the application window, the dialog box shown in Figure D53 is displayed. Once the list of the courses is displayed, the student can do the following things:

- Getting more information about the course highlighted in the list and displaying it in the "Course Info Dialog Box" shown in Figure D54, by pressing the "More Info" button.
- Printing the list of displayed courses by pressing the "Print List" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Course List Dialog Box" by pressing the "Help" button.

Double-clicking a course entry in the list will also bring more information about this course.



Course No	Course Name
CS0001	
CS0810	Thesis Research
CS2450	Computer Programming with FORTRAN
CS2920	Introductory Topics in Computer Science
CS2970	Structured Programming with ADA
CS2971	Introduction to Object Oriented Prog. with C++
CS2972	Object-Oriented Programming with ADA
CS3010	Computer System Principles
CS3030	Computer Architecture and Operating Systems
CS3050	Sofware Development for Combat Systems
CS3111	Principles of Programming Languages
CS3113	Introduction to Compiler Writing
CS3200	Computer Architecture
CS3300	Data Structures

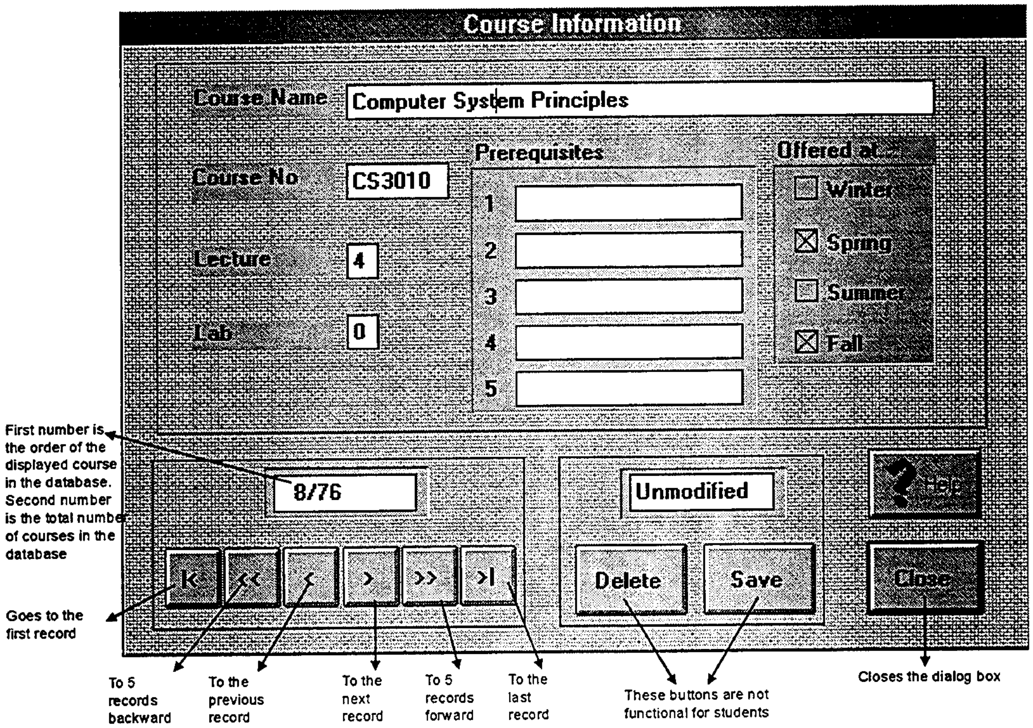
Figure D53. Course List Dialog Box

13. Course Information Dialog Box

If "More Info" command is selected from the "Student Matrix Dialog Box" or a course number is double-clicked in "Track Information Dialog Box", the dialog box shown in Figure D54 is displayed. Once the course information is displayed, the student can do the following things:

- Going to other course records by using the "Navigation Buttons".
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Course Information Dialog Box" by pressing the "Help" button.

It is important to note that, a course is not allowed to be deleted if it is a track requirement or a track elective for a track or a course in one of the default matrices. The user is first prompted to remove those courses from the tracks or default matrices found.



Course Information

Course Name: Computer System Principles

Course No: CS3010

Lecture: 4

Lab: 0

Prerequisites
1
2
3
4
5

Offered at
☐ Winter
☒ Spring
☐ Summer
☒ Fall

8/76

Unmodified

First number is the order of the displayed course in the database. Second number is the total number of courses in the database

Goes to the first record

To 5 records backward

To the previous record

To the next record

To 5 records forward

To the last record

These buttons are not functional for students

Closes the dialog box

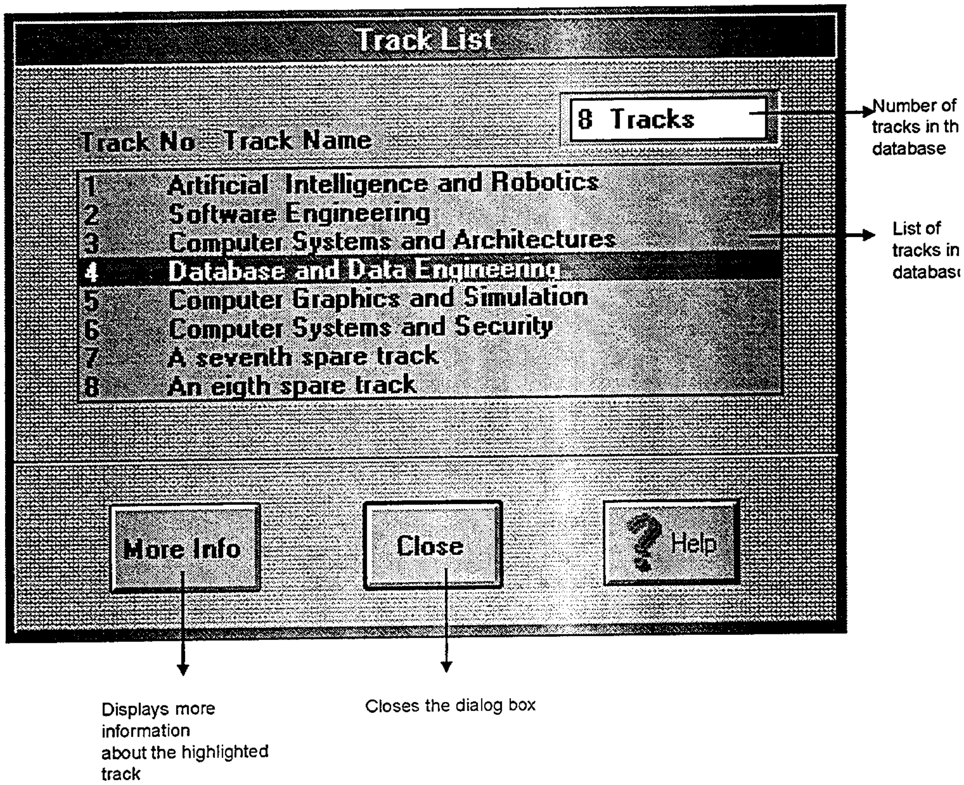
Figure D54. Course Information Dialog Box

14. Track List Dialog Box

If "Tracks" command is selected from the "Student Matrix Dialog Box" or from the menu or the speed bar in the application window or from some other dialog box, the dialog box shown in Figure D55 is displayed. Once the list of the tracks is displayed, the student can do the following things:

- Getting more information about the track highlighted in the list and displaying it in the "Track Info Dialog Box" shown in Figure D56 by pressing the "More Info" button.
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Track List Dialog Box" by pressing the "Help" button.

Double-clicking a track entry in the list will also bring more information about this track.



Track No	Track Name
1	Artificial Intelligence and Robotics
2	Software Engineering
3	Computer Systems and Architectures
4	Database and Data Engineering
5	Computer Graphics and Simulation
6	Computer Systems and Security
7	A seventh spare track
8	An eigth spare track

Number of tracks in th database

List of tracks in databas

Displays more information about the highlighted track

Closes the dialog box

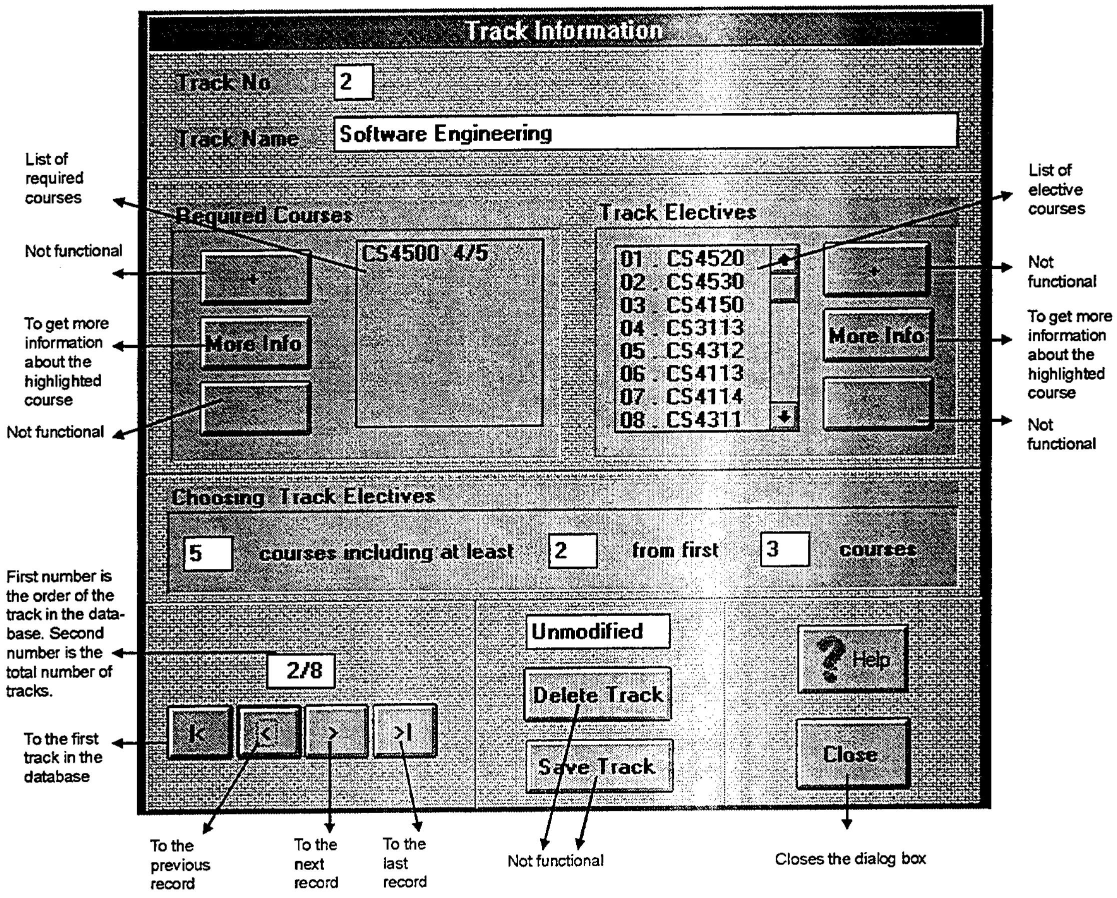
Figure D55. Track List

15. Track Information Dialog Box

If "More Info" command is selected from the "Student Matrix Dialog Box" or a track name is double-clicked in "Select Track" or "Change Track Dialog Box", the dialog box shown in Figure D56 is displayed. Once the track information is displayed, the student can do the following things:

- Getting information about the track requirements.
- Getting more information about a track requirement course by either pressing the "More Info button in "Required Courses" section or by double-clicking the course number in the "Required Courses" list.
- Getting more information about a track elective course by either pressing the "More Info button in "Elective Courses" section or by double-clicking the course number in the "Elective Courses" list.
- Going to other track records by using the "Navigation Buttons".
- Closing the dialog box by pressing the "Close" button.
- Displaying the on-line help for "Track Information Dialog Box" by pressing the "Help" button.

The rules of selecting the track electives are important to note for the students, because, if the students do not have the specified number of courses from the list of track electives in their matrices, they will not be allowed to save the matrices.



Track Information

Track No: 2

Track Name: Software Engineering

Required Courses

CS4500 4/5

Track Electives

01 . CS4520
02 . CS4530
03 . CS4150
04 . CS3113
05 . CS4312
06 . CS4113
07 . CS4114
08 . CS4311

Choosing Track Electives

5 courses including at least 2 from first 3 courses

Unmodified

2/8

List of required courses

Not functional

To get more information about the highlighted course

Not functional

First number is the order of the track in the database. Second number is the total number of tracks.

To the first track in the database

To the previous record

To the next record

To the last record

Not functional

Closes the dialog box

List of elective courses

Not functional

To get more information about the highlighted course

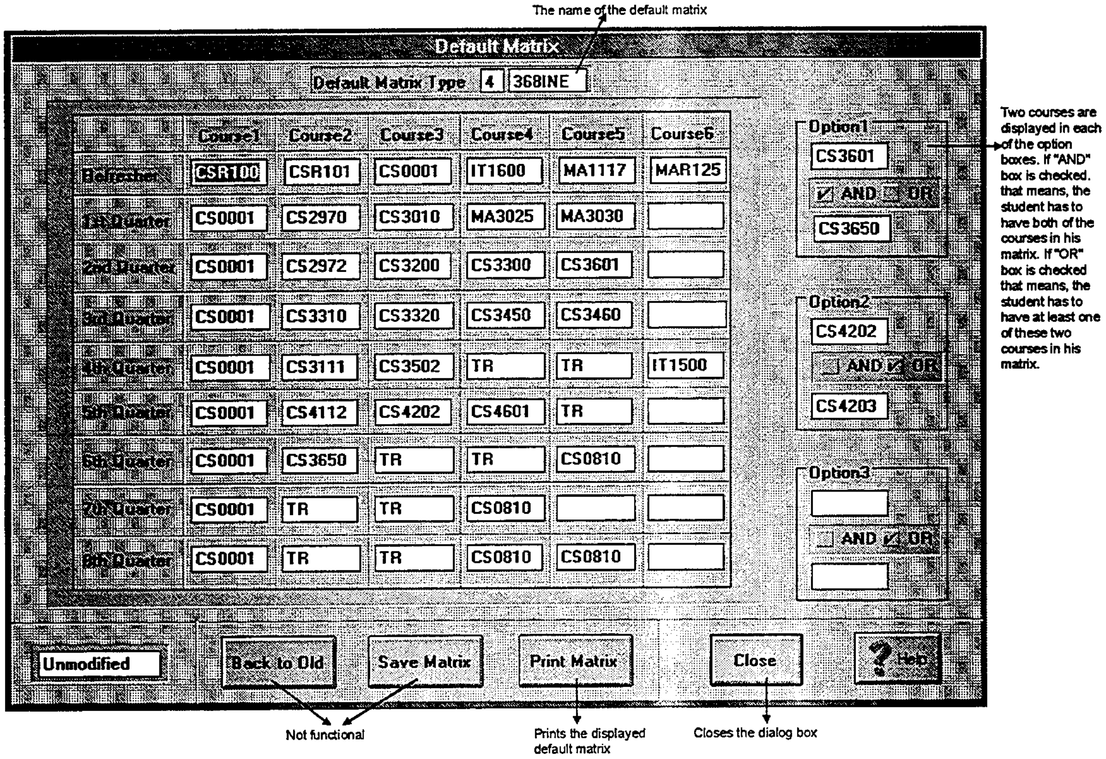
Not functional

Figure D56. Track Information Dialog Box

15. Default Matrix Dialog Box

If "Default Matrix" command is selected from the "Student Matrix Dialog Box" or from the menu or the speed bar in the application window, the dialog box shown in Figure D57 is displayed. Once the default matrices is displayed, the student can do the following things:

- Getting information about the default matrix courses.
- Printing the displayed default matrix by pressing the "Print Matrix" button.
- Closing the dialog box by pressing the "Close" button.
- Getting on-line help for "Default Matrix Dialog Box" by pressing the "Help" button.



Default Matrix Type 4 368INE

	Course1	Course2	Course3	Course4	Course5	Course6
Refresher	CSR100	CSR101	CS0001	IT1600	MA1117	MAR125
1st Quarter	CS0001	CS2970	CS3010	MA3025	MA3030	
2nd Quarter	CS0001	CS2972	CS3200	CS3300	CS3601	
3rd Quarter	CS0001	CS3310	CS3320	CS3450	CS3460	
4th Quarter	CS0001	CS3111	CS3502	TR	TR	IT1500
5th Quarter	CS0001	CS4112	CS4202	CS4601	TR	
6th Quarter	CS0001	CS3650	TR	TR	CS0810	
7th Quarter	CS0001	TR	TR	CS0810		
8th Quarter	CS0001	TR	TR	CS0810	CS0810	

Option1
CS3601
☑ AND ☐ OR
CS3650

Option2
CS4202
☐ AND ☑ OR
CS4203

Option3
☐ AND ☑ OR

Two courses are displayed in each of the option boxes. If "AND" box is checked. that means, the student has to have both of the courses in his matrix. If "OR" box is checked that means, the student has to have at least one of these two courses in his matrix.

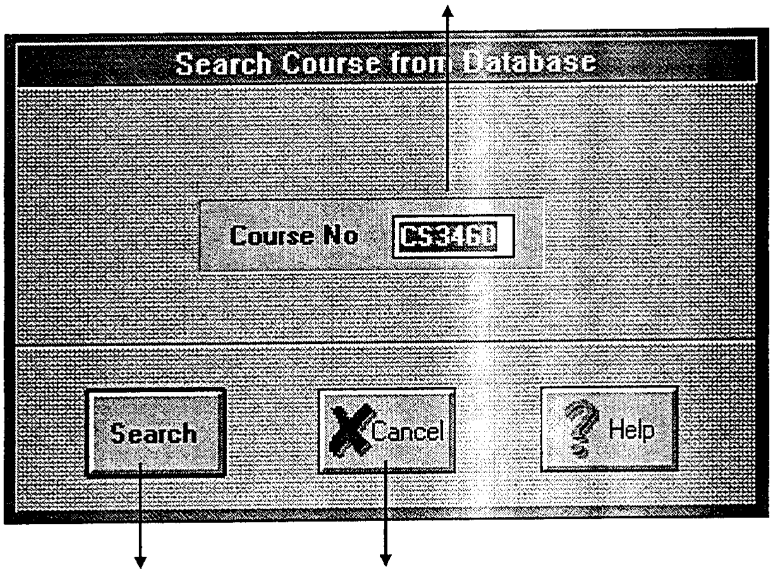
Figure D57. Default Matrix Dialog Box

16. Search Course Dialog Box

If "Search Course" command is selected either from the application menu or the speed bar, the dialog box shown in Figure D58 is displayed. In order to be able to search a course successfully, the entered "Course No" has to be valid. If the course is found in the database, the course information is displayed in "Course Information Dialog Box" shown in Figure D54.

Pressing the "Cancel" button will close the dialog box and pressing the "Help" button will display the on-line help for the "Search Course Dialog Box".

Course No of the student to be deleted from the database.



Course No CS3460

Searches the course in database whose Course No is entered

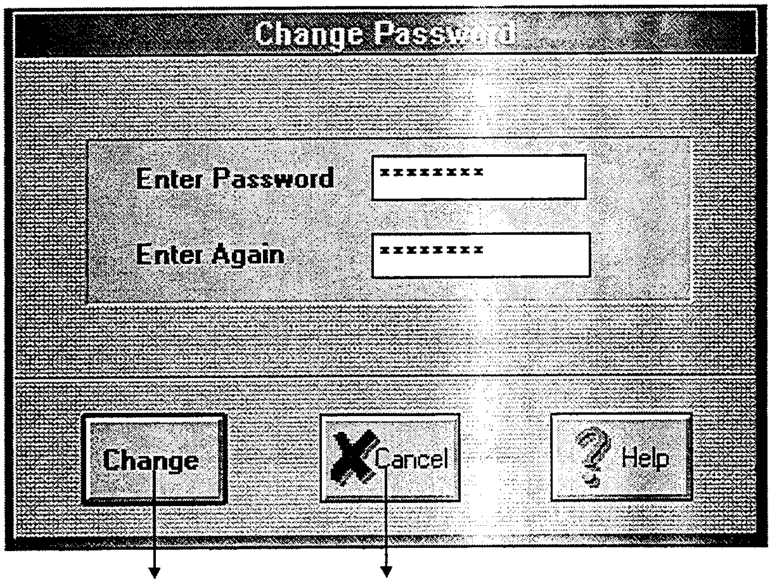
Closes the dialog box without making any search.

Figure D58. Search Course Dialog Box

17. Change Password Dialog Box

If "Change Password" command is selected from the "Student Matrix Dialog Box", the dialog box shown in Figure D59 is displayed. In this dialog box, the student can do the following things:

- Entering the new password twice and changing the his old password to the newly entered one by pressing the "Change" button. The new password has to be entered twice to make sure that he is actually entering what he is intending to, since the entered password is not displayed on the screen. If the entered two passwords are not the same, he will be prompted to enter it again.
- Closing the dialog box without changing the password by pressing the "Cancel" button.
- Displaying the on-line help for "Change Password Dialog Box" by pressing the "Help" button.



Enter Password xxxxxxxx

Enter Again xxxxxxxx

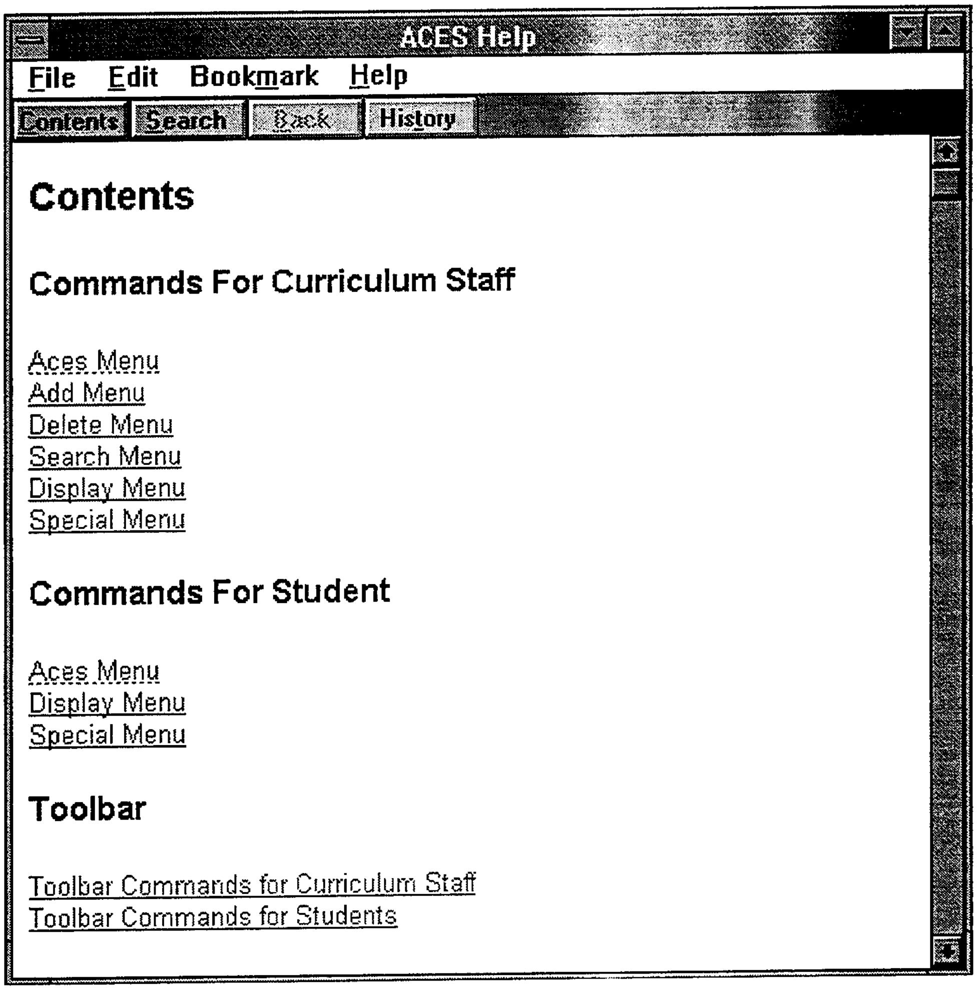
Changes the password of the student to the newly entered one

Closes the dialog box without changing the password

Figure D59. Change Password Dialog Box

18. On-Line Help

If "Help Contents" command is selected either from the application menu or the speed bar, the help window shown in Figure D60 is displayed. This help window is a typical on-line help for "Windows Environment". There are hyper-links for the words or phrases written in green which means that clicking such a word or phrase will display another page with the new information about the selected word or phrase. It is possible to move back through the page links by pressing the "Back" button in the speed bar. For more information about how to use the on-line help, the "Using Help" command should be chosen from the application menu.



Contents

Commands For Curriculum Staff

Aces Menu
Add Menu
Delete Menu
Search Menu
Display Menu
Special Menu

Commands For Student

Aces Menu
Display Menu
Special Menu

Toolbar

Toolbar Commands for Curriculum Staff
Toolbar Commands for Students

Figure D60. Help Contents Window

LIST OF REFERENCES

1. Borland C++ User's Guide, pp 83-85.
2. Borland C++ User's Guide, pp 25-26.
3. Borland Object Windows for C++ Reference Guide, pp 4.
4. Christian Kaare, Borland C++ Techniques and Utilities, 1993, pp 280.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   Cameron Station
   Alexandria, Virginia 22304-6145

2. Library, Code 052 2
   Naval Postgraduate School
   Monterey, California 93943-5101

3. Prof. Man-Tak Shing, Code CS/Sh 4
   Department of Computer Science
   Naval Postgraduate School
   Monterey, California 93943

4. LCDR Frank Kelbe, Code CS/Ke 1
   Department of Computer Science
   Naval Postgraduate School
   Monterey, California 93943

5. Ted Lewis, Code CS/Lt 1
   Department of Computer Science
   Naval Postgraduate School
   Monterey, California 93943

6. Curricular Officer, Code 32 2
   Naval Postgraduate School
   Monterey, California 93943

7. CDR Thomas Hoskins 1
   Department of Computer Science
   Naval Postgraduate School
   Monterey, California 93943

8. Erkan Ertugrul 2
   Plevne Mah. Nur Sok.
   No: 33/6
   Sincan/ Ankara TURKEY

9. Kara Harp Okulu Kutuphanesi 1
   Bakanliklar/Ankara TURKEY